Social Networks: Community Detection

Pavla Dráždilová

Doctoral Thesis

Department of Computer Science
Faculty of Electrical Engineering and Computer Science
VŠB – Technical University of Ostrava
Ostrava 2012

Abstract

In this thesis, we focused on finding communities in complex networks via two different ways. The first approach is realized by specifying a subnet with the context of selected terms which determine the selected subnet. The second approach to finding communities is realized by our proposed algorithm Left-Right-Oscillate (LRO), which is based on spectral ordering of graph vertices. These two approaches allow us to detect a desired community – either by the size of the smallest communities, or by the level of modularity and by selected terms – in large networks such as DBLP.

Keywords

Spectral clustering, spectral ordering, community detection, discovering social network, modularity.

Abstrakt

V této dizertační práci jsme se zaměřili na hledání komunit v komplexních sítích dvěma odlišnými způsoby. První přístup je realizován pomocí určení podsítě v kontextu k vybraným termům, které nám určují charakter vybrané podsítě. Druhý přístup k hledání komunit je realizován pomocí námi navrženého algoritmu Left-Right-Oscilate (LRO), který je založen na spektrálním uspořádání vrcholů grafu. Tyto dva přístupy nám umožňují detekovat požadované komunity – buď podle velikosti nejmenších komunit nebo úrovně modularity nebo podle vybraných termů – v rozsáhlých sítích jakou je například DBLP.

Klíčová slova

Spektrální shlukování, spektrální uspořádání, detekce komunit, nalezení sociální sítě, modularita.

Acknowledgement

I would like to thank Václav Snášel for his leadership, knowledge and skills, and Jan Martinovič and Kateřina Slaninová for skilled help during my PhD study.

Author:
Pavla Dráždilová
pavla.drazdilova@vsb.cz
http://www.cs.vsb.cz/drazdilova

Thesis supervisor:
Václav Snášel
vaclav.snasel@vsb.cz
http://www.cs.vsb.cz/snasel

Department of Computer Science
Faculty of Electrical Engineering and Computer Science
VŠB – Technical University of Ostrava
17. listopadu 15, 708 33 Ostrava–Poruba
Czech Republic
http://www.cs.vsb.cz
http://fei.vsb.cz
http://www.vsb.cz

Contents

List of Figures

List of Tables

CHAPTER I

Introduction

The internet today provides several linked and diverse methods of interaction that, at first, may seem user unfriendly and difficult to describe or evaluate. Our goal is to better understand a relationship between objects that interest us and which create a network structure. Therefore, when creating new relations between objects, we take into consideration the original information we have on relations between objects, or we create new relations based on shared activities [64, 65]. Then we can present the original data collection in a bipartite graph, capturing the relationship between objects and activities carried out by a given object (shared projects, completed tasks). Newly created relations include more information about the object acquired from the original data collection for more complex and precisely described relations between the two objects.

In this thesis, we will use several data collections, obtained from various areas of the internet, to create new synthetic relations that will address these shortcomings (e.g. e-learning system Moodle[1], EDISON - Education Information System on Net[2], network of co-authors DBLP[3], CodePlex[4]). Then, we will use these relations to create a network to describe the original data in a new way (more details were published in our papers [32, 79, 93]).

In our research project, these newly created and detailed network descriptions are concerned with user relationships. In this thesis, we focused on finding communities in complex networks by two different ways. The first approach is realized through specification of a subnet with the context of selected terms which determine the selected subnet. The second approach to finding communities is realized by our proposed algorithm Left-Right-Oscillate (LRO), which is based on spectral ordering of graph vertices. Another point of interest is to determine the structure of the communities in the networks.

[1] http://http://moodle.org/
[2] https://edison.vsb.cz/
[3] The DBLP Computer Science Bibliography - http://www.informatik.uni-trier.de/ ley/db/
[4] http://www.codeplex.com/

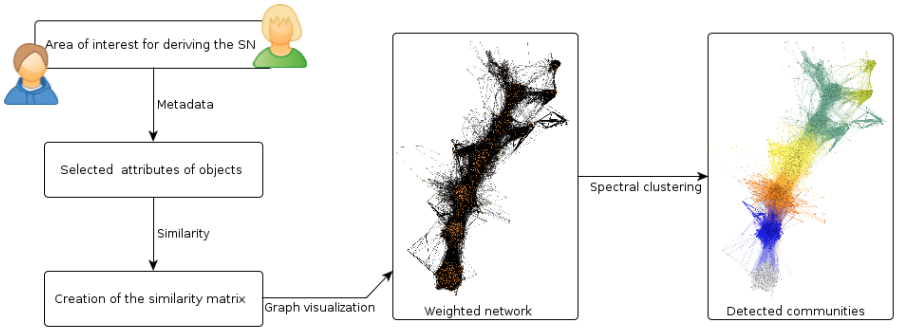

Figure 1.1: Process of network deriving

Aside from creating an evaluated network that describes a given area of interest, we also focused on other ways to specify an appropriate area of interest. This pertains to a context determination for individuals in order to properly select a terminology. Terms are extracted from an accessible object signature (e.g. in DBLP, it refers to the title of an article), to specify more accurately a user group of interest which, on the other hand, creates the network within the context of a given term. This activity of ours is inspired by the need to identify a user group occupied with a given issue. The goal is to create a support system for searching information (i.e. at the start of research in a new field) that more clearly specifies a selected term (e.g. the term "state-of-the-art"). Using our newly-designed methods, we are able to find various user communities involved in a given field, as well as their mutual relationships.

For searching communities in a large network, we based our work on the method of spectral clustering to efficiently find a cluster of objects that, in our data collection, represents users with similar behaviour patterns, interests and/or the same activities. This method was expanded to include overlapping clustering, a method that employs a much more natural process for capturing individual communities of users within a network. Obviously, users may be involved in several hobbies or activities, and may belong to various "groups of interest". Our goal is not to determine just one community to which a user belongs but also to determine which communities the user is involved in.

Thesis Outline

In the first part of this paper, we provide an overview of social networks and options for creation of a new way for evaluation of relations between the objects. Here, we present a summarized definition of similarities and distances that we can use to create a new way of evaluating user relations. For an expanded evaluation of user relations, we have decided to define "context" as the relation between the selected terms and the user who used it.

In the second part of this thesis, we focus on spectral clustering method. We provide an overview of clustering algorithms aimed at seeking out communities within social networks. The core of this work is based on spectral ordering. Spectral ordering is the first part of an algorithm used to seek out communities within selected, evaluated networks. More precise designations for communities

are then monitored using modularity. Our Left-Right-Oscillate algorithm is described in details with the motivation that led us to the construction of the algorithm which increases the initial modularity.

The third part of this thesis is experimental. Here, we describe various data collections and their role in the creation of newly evaluated relations and in searching for communities in a newly evaluated network. Our algorithm detects communities and enables us to determine the natural amount of communities within a network (eliminating the need to specify the amount of communities within that network as k-mean clustering method). Furthermore, there are the results of the LRO algorithm that demonstrates its properties and properties of obtained communities.

CHAPTER II

Social Network

Social networking is a complex, large and expanding sector of the information economy. Researchers' interest in this field is growing rapidly. It has been studied extensively since the beginning of the 20th century. The first normative contributions in this area were proposed in 1970s by sociologist Mark Granovetter and mathematician Linton C. Freeman. The basic theory "The Strength of Weak Ties" was mentioned in 1973 [45]. Granovetter argued that within a social network, weak ties are more powerful than strong ties. Another significant principle was published in 1979 by Linton C. Freeman [41]. In his work, there was presented definition of centrality, which is one node's relationship to other nodes in the network. He defined basic metrics like degree, control and independence, from which reason researchers proceed in their present works.

Social network researchers have acquired data for their studies using various methods. In the past, these studies were only based on questionnaire data, which typically reached the amount of hundreds individuals [101]. In the late 1990s, new technologies such as internet and cellular phones enabled researchers to construct large scale networks using emails [35], phone records [81] or web search engines [57].

Social network (SN) is a set of people or groups of people with similar patterns of contacts or interactions such as friendship, co-working, or information exchange [42]. Citation networks, human activity on the internet (email exchange, consumer behaviour in e-commerce), physical and biochemical networks are some examples of social networks. Social networks are usually represented by using graphs, where nodes represent individuals or groups and lines represent contacts among them. The configuration of relations among network members identifies a specific network structure, and this structure can vary from isolated structures where no members are connected to saturated structures in which everyone is interconnected.

Social network analysis was defined by Barry Wellman as "work at describing underlying patterns of social structure, explaining the impact of such patterns on behaviour and attitudes" [54, 103]. Therefore, researchers are not interested only

on describing the different social structures, but they emphasize on investigating the consequences of this variation on the member's behaviours.

2.1 Related Work

Social network analysis can be very useful and applicable in many spheres. For example, in the commercial sphere, it is used to conduct viral marketing to explore relations between existing and potential customers for increasing sales of products and services. It is used in biological and medical diagnostics for application of viral prevention. In law enforcement, knowledge of social networks can be useful in criminal investigations concerning organized crimes (terrorism, money laundering, drugs, etc.). With an increasing amount of people using mobile phones, mobiles have become another area of research. For example, information of social network access that is obtained from call logs is presented in the paper [83]. In this work, an end-to-end system for inferring social networks based on call logs using kernel-based naive Bayesian learning is proposed.

Due to the growth of the Internet, online advertising markets and other web services, such as recommendation systems and auction markets in e-business sphere are continually developed. In e-commerce, web adaptation on the basis of users' behaviours or online supply of goods and services coming from previous purchases is used on the basis log analysis, webs, and social networks.

Recently, many researchers have focused on analysing the growth of social communities in the Internet world. We can observe occurrence and great expansion of various social bookmarking systems based on recommendations and sharing of various types of information like URLs (del.icio.us[1]), multimedia files (photos, videos, music – Flickr[2], YouTube[3]), blogs (MySpace[4], LiveJournal[5], citation webs), etc. The structure behind these social systems (called folksonomies) can be represented as a collection of users, tags and resource nodes. These collections of data can be viewed using graphs or visualization software and can be analysed with orientation to structural properties to show the growth and exploration of social networks.

2.1.1 Online Social Networks

Web mining techniques used for online social network analysis are similar to other mining techniques. There are many traditional techniques used such as classification, clustering, association rule mining etc. For data interpretation and analysis of results, visualization techniques like graphs are usually used. Web mining techniques can be divided (according to the analysis target) to web content mining (text data and natural language processing, analysis focused on other multimedia sources, semantic web, group analysis), web structure mining oriented to structure of websites (implementation of crawlers and Deep-web

[1] http://delicious.com/
[2] http://www.flickr.com/
[3] http://www.youtube.com/
[4] http://www.myspace.com/
[5] http://www.livejournal.com/

research), and web usage mining focused on how websites are used (clickstream data analysis, navigation behaviour, and recommendation systems).

Many researchers focus on identifying and analysing community structures in networks growing from web users. Analysis of topological characteristics of the tripartite hyper graph of queries, users, and bookmarks on a large snapshot of del.icio.us web site and on query logs of two large search engines is described in the article [56]. The extensive analysis of characteristics of large online social network MySpace was published ([11]). This study was oriented to the sociability of users based on relationship, messaging, and group participation, and on the demographic characteristics of users with emphasis on the correlation of their privacy references and on text analysis with intention to construct language models used by MySpace users. In the paper [59], researchers analysed 70 large sparse real-world networks and defined network community profile plot, which characterizes the "best" profile community. They compared and contrasted several methods to optimize the calculations based on conductance measurement.

Another study [22] is focused on the analysis of social networks in relation to similarity and social ties. Authors developed techniques for identifying and modelling interactions between social influence and selection and consider the relative value of similarity and social influence in modelling future behaviour of social network. The problem of defining proximity measure between groups (communities) in online social networks is presented in the article [88]. This measure is used in recommendation systems to help the users in selecting the groups of their interest.

2.1.2 Extraction of social networks

One area where we can obtain information on various structured Social networks (SN) is the internet. Several studies have addressed extraction of social networks automatically from various sources of information such as the Web, e-mail, and contacts [106, 63, 62, 1, 23]. While most approaches for social network extraction have focused on the strength of the relation, few studies have addressed automatic identification of underlying relations. Matsuo et al. employed a supervised machine learning method to classify four types of relations in a research community [62]. Pattern-based approaches [19] seek phrases or sentence structures that explicitly show relations between instances. However, most Web documents have a very heterogeneous structure, even within individual web pages. Therefore, the effectiveness of the pattern-based approach depends on the domain to which it is applied.

We can use knowledge extraction of social network for experts finding. The task of experts finding focuses on finding persons with high level of experience on a specific topic. To achieve this objective, researchers approached this task mainly in three different ways. The first group applied information retrieval techniques to solve it [25], the authors of this paper proposed a weighted language model, which introduces a document prior probability to measure the importance of the document written by an expert. The second group approached the task of experts finding using social network analysis metrics [108], in this study a large online help seeking community, the Java Forum, was analysed using social network analysis methods and a set of network-based algorithms, including PageRank and HITS. While the third group used a hybrid approach of information retrieval and social network analysis for finding academic experts

[109, 17], in the paper [109] the authors created a local information document for each person to measure his initial level of experience on a topic using information retrieval models then they applied propagation on the graph of experts to update his level of expertise according to his relations with the other nodes, and in the article [17] the authors proposed an Authoritative Expert Finding System, called AEFS, which ranks the publications of experts to indicate their level of expertise and removes duplicated citations using the concept of social network centrality.

2.1.3 Evaluation of Social Networks

For a description of SNs defined in 1979, see Linton Freeman [41] various *centrality*, where individual network nodes are directly evaluated, or where the average value of selected centrality in a graph may be an item of interest.

A primary use of graph theory in social network analysis is to identify the important or prominent actors at both the individual and group levels of analysis. *Centrality* and *prestige* concepts and measures seek to quantify graph theoretic ideas about an actor's prominence within a complete network by summarizing the structural relations among all nodes. Centrality is when a prominent actor has high involvement in many relations, regardless of whether sending or receiving ties. Prestige is when a prominent actor initiates few relations but receives many directed ties. Knoke and Yang defined the above mentioned terms (see [54]).

For the determination of actor *degree centrality* require applications of the matrix algebra notation. Unlike actor degree centrality, group degree centralization measures the extent to which the actors in a social network differ from one another extent to which the actors in a social network differ from one another in their individual degree centralities.

Actor *closeness centrality* was developed to reflect how near a node is to the other nodes in a social network [87]. Closeness and distance refer to how quickly an actor can interact with others, for example, by communicating directly or through very few intermediaries. An actor's closeness centrality is a function of its geodesic distance (length of the shortest path connecting a two nodes) to all other nodes.

Authors in the paper [80] combined existing methods on calculating exact values and approximate values of closeness centrality and presented new algorithms to rank the top-k vertices with the highest closeness centrality.

Betweenness concept of centrality concerns how other actors control or mediate the relations between two nodes that are not directly connected. Actor betweenness centrality measures the extent to which other actors lie on the geodesic path between pairs of actors in the network.

Brandes in the article [8] presented a faster algorithm for betweenness centrality focused on large, yet very sparse networks. The algorithms is based on a new accumulation technique that integrates well with traversal algorithms solving the single-source shortest-paths problem, and thus exploiting the sparsity of typical instances.

Prestige is defined as the extent to which a social actor in a network "receives" or "serves as the object" of relations sent by others in the network. The sender-receiver or source-target distinction strongly emphasizes inequalities in control

over resources, as well as authority and deference accompanying such inequalities [53].

And for centrality in Egocentric Networks with g alters is defined ego i's actor degree centrality as the maximum possible value of actor degree centrality, g-1.

One of measure in networks is the global *clustering coefficient*. The local clustering coefficient of a vertex in a graph is based on ego's network density or local density. Duncan J. Watts and Steven Strogatz introduced the measure in 1998. And the global clustering coefficient is concerned with the density of triplets of nodes in a network. A triplet can be defined as three nodes that are connected by either two (open triplet) or three (closed triplet) ties. A triangle consists of three closed triplets, each centred on one node. The global clustering coefficient is defined as the number of closed triplets over the total number of triplets. In the article [82], authors proposed a generalization of this coefficient that retains the information encoded in the weights of ties.

To understand networks and their participants, we provide the location of actors in the network. Measuring the network location is finding the centrality of a node. These measures determine the various roles and groupings in a network – who are the connectors, specialists, leaders, bridges, isolates, where are the clusters and who is in them, who is in the core of the network, and who is on the periphery.

Another source that describes the quality of clusters in networks is modularity. This concept will be defined in the section 3.3.5. We use this concept to gauge quality in clusters we find. Our goal is to verify that the algorithms we use improve the quality in found communities (that modularity values grow).

Recent interest in scale-free networks started in 1999 with work of Barabasi and Albert [5], who mapped the topology of a part of the World Wide Web, finding that some nodes, which they called *hubs*, had many more connections than others, and that the network as a whole had a power-law distribution of the number of links connecting to a vertex. A *scale-free network* is a network whose degree distribution follows a *power law*, at least asymptotically. That is, the fraction $P(k)$ of nodes in the network having k connections to other nodes goes for large values of k as $P(k)\tilde{c}k^{-\gamma}$ where c is a normalization constant and γ is a parameter whose value is typically in the range $2 < \gamma < 3$, although occasionally it may lie outside these bounds. *Power law* graphs are random graphs specified by a power law degree distribution $Pr[D = k] = L(k)k^{-\gamma}$, where $L(k)$ is a slowly varying function of k [70]. The power law degree distribution is followed by many natural and artificial networks such as the scientific collaborations, the world-wide web, and the internet.

In addition to degree distribution of network we can determine the distribution of components and communities in the network.

One of the most remarkable and widely discussed of network phenomena is the *small-world effect*, the finding that in many networks the typical network distances between vertices are surprisingly small. The small-world model proposed by Watts and Strogatz [102] encompasses the following two structural features as observed in real-world networks. Any two nodes can be reached within a small number of links despite the large size of networks. Nodes are well clustered in the sense that two direct neighbours of a node are more likely to be connected compared to those in random graphs.

2.2 Visualization of Social Networks

Creating visual images of networks can serve important heuristic purpose; visual images are powerful complements to quantitative analyses. Network images supplement statistical analyses and allow the identification of groups of people for targeting, the identification of central and peripheral individuals, and the clarification of the macro-structure of the network.

All graphs and also social networks can be represented as node-link graphs or as adjacency matrices. While graphs present visualizations of social networks, matrices use mathematical algebraic representations of network relations. Both this methods have their advantages and disadvantages. Graphs provide better visual illustrations of network structures but do not support mathematical manipulations. On the other hand, matrices are less user-friendly, but they facilitate sophisticated mathematical and computer analyses of social network data.

Networks can be arranged on a map to represent the geographic distribution of a population. Alternatively, algorithmically generated layouts have useful spatial properties: a force-directed layout can be quite effective for spatially grouping connected communities, while a radial layout intuitively portrays network distances from a central actor. Colour, size, and shape have been used to encode both topological and non-topological properties such as centrality, categorization, and gender. In recent years, such approaches have been effectively used in the analysis of domains such as e-mail communication [38], early online social networks [2], and co-authorship networks in scientific publications [44, 73, 3]. There are a number of systems for generating social network visualizations and performing statistical analyses for the purpose of sociological research, such as UCINet [6], Pajek [7], Vizster [8] [47] and NetMiner [9].

2.2.1 Node-link Diagram or Matrix Representation of Network

The majority of network visualization systems use the node-link representation. This representation is well suited to show sparse networks, but social networks are known to be globally sparse and locally dense. Therefore, social network visualization faces a major challenge, obtaining a readable representation for both the overall sparse structure of social network and its dense communities.

Once a possibility of hybrid visualization is MatLink [49], an enhanced matrixbased graph visualization that overlays a linear node-link diagram on the edges and adds dynamic feedbac of relationship between nodes.

Other a possibility is NodeTrix [50]. This representation integrates the best of two traditional network representations: node-link diagrams and adjacency matrix-based representations. The strength of this representation for analysing social networks is in combining the familiarity of node-link diagrams to under-

[6] http://www.analytictech.com/
[7] http://pajek.imfm.si
[8] http://hci.stanford.edu/jheer/projects/vizster/
[9] http://www.netminer.com/

stand the global structure of the network with the readability of matrices for detailed community analysis.

2.2.2 Clustered Graph Representation

Clustered graph representations reduce the visual complexity of graphs by grouping nodes, and are well suited to social networks as they highlight important structures, like communities and central actors linking them. Authors of MatLink and NodeTrix used NodeTrix for representation of clusters as visual adjacency matrices - each node is placed as a column and row in matrix and links between nodes are marked in the matrix. They improved the readability of clustered social networks using duplication of nodes [48].

Social analysts often need to adjust the level of clustering. Eades et al. proposed several solutions to draw clustered graphs [33], eventually showing several levels of clustering at the same time [34].

Other possibility, how to visualize clustered graph (hierarchical structures), is Treemap [10]. It is very effective in showing attributes of leaf nodes using size and color coding. Treemap enables users to compare nodes and sub-trees even at varying depth in the tree, and help them spot patterns and exceptions.

2.2.3 Software for visualization

The function of the software for social network visualization make complicated types of analysis and data handling transparent, intuitive, and more readily accessible. We can visualize a whole system or subsystem to explore the architecture to apply visual data mining, visual analytics techniques for defect discovery or manually discover anomalies.

We use Gephi[11] in our work for the network visualization. Gephi [6] is an open-source network analysis and visualization software package written in Java on the Netbeans platform. Gephi has been selected for the Google Summer of Code in 2009, 2010 and 2011.

Gephi has been used in a number of research projects in the university, journalism and elsewhere, for instance in visualizing the global connectivity of New York Times content [58] or mapping dynamic conversation networks on Twitter [10].

2.3 Distances and Similarities

In this section, we define the distances or similarities used for establishing weighted relationship between individuals. We use definitions from [26].

Definition 2.1 *Let X be a set. A function $d : X \times X \to \mathbb{R}$ is called distance (or dissimilarity) on X if, for all $\mathbf{x}, \mathbf{y} \in X$, the following holds:*

- $d(\mathbf{x}, \mathbf{y}) \geq 0$ *(non-negativity)*

[10] http://www.treemap.com/
[11] http://gephi.org/

- $d(\mathbf{x}, \mathbf{y}) = 0 \iff \mathbf{x} = \mathbf{y}$ *(identity of indiscernibles)*
- $d(\mathbf{x}, \mathbf{y}) = d(\mathbf{y}, \mathbf{x})$ *(symmetry)*

Definition 2.2 *A distance space is an ordered pair (X, d) where X is a non-empty set and d is a distance on X.*

2.3.1 Distances - Dissimilarities

A distance (or dissimilarity) between objects (which are represented by vector) $\mathbf{x}$ and $\mathbf{y}$ is function $d(\mathbf{x}, \mathbf{y}) : X \times X \to \mathbb{R}$ from the definition of distance. When we require triangle inequality $d(\mathbf{x}, \mathbf{y}) + d(\mathbf{y}, \mathbf{z}) \geq d(\mathbf{x}, \mathbf{z})$ for all $\mathbf{x}, \mathbf{y}, \mathbf{z} \in X$ then the function d is metric on X.

Some clustering algorithms use a dissimilarity matrix A (distance-space methods), where $a_{ij} = d(\mathbf{x_i}, \mathbf{x_j})$ for $\mathbf{x_i}, \mathbf{x_j} \in \mathbb{R}^n$. For example k-mean, single or average linkage method and other.

Some mostly used dissimilarity measures:

- Manhattan distance (L_1) $d(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{n} |x_i - y_i|$
- Euclidean distance (L_2) $d(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$
- Minkowski distance (L_q for $q \geq 1$) $d(\mathbf{x}, \mathbf{y}) = \sqrt[q]{\sum_{i=1}^{n} |x_i - y_i|^q}$
- Chebychev distance (L_∞) $d(\mathbf{x}, \mathbf{y}) = \max_{i=1\ldots n} |x_i - y_i|$

2.3.2 Similarities

Definition 2.3 *Let X be a set. A function $s : X \times X \to \mathbb{R}$ is called similarity (or proximity) on X if s is non-negative, symmetric, and if $s(x, y) \leq s(x, x)$ holds for all $x, y \in X$, with equality if and only if $x = y$.*

Main transforms used to obtain a distance (dissimilarity) d from a similarity s are: $d = 1 - s$, $d = \frac{1-s}{s}$, $d = \sqrt{1-s}$, $d = \sqrt{2(1-s^2)}$, $d = -\ln s$ and $d = \arccos s$, where $s \in < 0, 1 >$

Definition 2.4 *Vector space model (or term or attribute vector model) is an algebraic model for representing objects (text documents, users, developers) as vectors of attributes (identifiers, terms), such as, for example, index terms (user's articles, developer's projects).*

This approach allows easy data handling, since object similarities are expressed between representations in the vector space. Vector model is used in information filtering, information retrieval, indexing and relevancy rankings.

We use similarities for evaluation of relation between two objects which are described by vectors $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$.

- Cosine similarity: $s(\mathbf{x}, \mathbf{y}) = \frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_{i=1}^{n} x_i^2} \sqrt{\sum_{i=1}^{n} y_i^2}}$
- Tanimoto coefficient:

$$s(\mathbf{x}, \mathbf{y}) = \frac{\sum_{i=1}^{n} x_i y_i}{\sum_{i=1}^{n} x_i^2 + \sum_{i=1}^{n} y_i^2 - \sum_{i=1}^{n} x_i y_i}$$

- Ruzicka similarity: $s(\mathbf{x}, \mathbf{y}) = \frac{\sum_{i=1}^{n} min\{x_i, y_i\}}{\sum_{i=1}^{n} max\{x_i, y_i\}}$

2.4 Discovering social networks

Today, the most widely used SN is Facebook, was launched in February 2004 and has more than 845 million monthly active users and 483 million daily active users on average in December 2011. Youngest SN, which was launched on June 28, 2011, is Google+ with 98 million users. 60 % users engage daily, and over 80% weekly. Twitter doesn't share this number. These online SNs provide a wide range of service options that simplify communication, sharing information, people searching, etc.

Relations in on-line SNs are presented directly to users who specify their own relation to other users. SN users provide a wide range of personal information, allowing them to be easily found by users with similar interests, hobbies, professional specializations and more.

In the data collections that we have been dealing with, user relations have not been precisely determined and evaluated. For example, data from DBLP, an index that lists a network of co-author publications, present user relations if they have collaborated on an article. This relation allows for an evaluation of the intensity with which the authors collaborated on a given project, but does not capture other properties that would aid deducing the legibility of the terminology used. Despite the fact that researched networks did not all necessarily fall under the category of social networks, we attempted to discover and evaluate relations among their users based on shared activities.

2.4.1 Deriving relations

Van der Aalst at al. in [96] derived relations from event logs. They derived these types of metrics from event logs:

- *metrics based on possible causality* monitor for individual cases how work moves among performers. Within a case (i.e., process instance) there is a handover of work from individual i to individual j if there are two subsequent activities where the first is completed by i and the second by j.

- *metrics based on joint cases* ignore causal dependencies but simply count how frequently two individuals are performing activities for the same case. If individuals work together on cases, they will have a stronger relation than individuals rarely working together.

- *metrics based on joint activities* do not consider how individuals work together on shared cases but focus on the activities they perform. People doing similar things have stronger relations than people doing completely different things.

- metrics based on special event types consider the type of event. Van der Aalst assumed that events correspond to the execution of activities. For example, if i frequently delegates work to j but not vice versa it is likely that i is in a hierarchical relation with j. These observations are particularly interesting in SNA since they represent explicit power relations.

Another option would be to create a one-mode graph from a bipartite graph [111], where the bipartite graph captures relations between to different types of groups (e.g. developers and their current projects, or students and exercise classes they attend). These user relations would then be evaluated measuring the intensity of their shared activity. We have added context obtained from a data collection using term extraction [65] for the evaluation of relations ([65]).

2.4.1.1 Log analysis

We have implemented several methods for deducing user relations. Individual methods are dependent upon the data collections that we are working with. Otherwise, we would have to search for user relations in log files (Moodle), as well as in data describing developer activities in various projects.

Our approach to deducing relations between objects has been inspired by the aforementioned van der Aalst method. Some of our data collections contain logs recording user activities in system (Moodle collection). We have tested various methods for evaluating similarities among these users, including cosine similarity in sequenced logs within articles [61], suffix tree applications in other articles [95], and searching for the longest common substring in [94].

2.4.1.2 Relations between Persons on the Basis of Term Context

Another method that we have used for more precise evaluation of the intensity of person's relations was to ascertain the context among persons (e.g. authors or developers) and the terminology they used (for example in article titles in DBLP or project descriptions in Codeplex).

In [12] authors present a novel probabilistic topic model to analyze text corpora and infer descriptions of its entities and of relationships between those entities. [100] presents topical n-grams, a topic model that discovers topics as well as topical phrases. In [89] authors describe how they used FCA to create a visual overview of the DBLP scientific journals classification based on their aims and scopes. Another area that utilizes text information is finding of expert in DBLP bibliography data [25], or the analysis of communities based on DBLP [7].

We use in our approach terms for evaluation of the relation between persons (co-authors or developers). We extend standard evaluation of the relation, which is based on the number of the common projects (publications, articles or active projects), by a factor that represent context between person and term selected from the term set.

Term set is understood as a collection of all keywords, which are extracted from the input text. For the DBLP as the input texts were used titles of articles and in Codeplex were used descriptions of projects. A detailed description of *term set* was published in our article [65].

2.4.1.3 Relations between Persons

Besides the computation of evaluated term set itself, we can compute *association strength* between the two persons. This method is not only interesting by itself, but it is also essential for extended evaluation of the term list by selected context. *Relevancy* between persons is based on the participation on the same project. This relevancy is then approximated by Jaccard coefficient [26].

Let A be a set of all persons in dataset. We define a single person A_i. For A_i, it is evaluated the strength of association with the other persons (co-participants). The strength of participation could be computed in a way that we go through all the person's projects while marking all the participated persons. Afterwards, we can order them by frequency of common projects. The set of co-participants of person A_i is marked as C_{A_i}. Let set P be a set of all projects and P_{A_i} be a set of all projects of person A_i.

The *association strength* between the persons A_i and A_j can be defined with Jaccard coefficient that reflects mainly the proximity of both persons from number of their common projects:

$$Q(A_i, A_j) = \frac{|P_{A_i} \cap P_{A_j}|}{|P_{A_j}| + |P_{A_i}| - |P_{A_i} \cap P_{A_j}|} \tag{2.1}$$

If this method is applied to all the persons, we obtain weighted undirected graph that can be considered as a synthetic social network (with re-weighted edges between persons). This approach was inspired by [29].

2.4.1.4 Persons and the Term Context

If we define a set T as the set of all terms in the input text and T_{A_i} as the set of all the terms that could be found in the projects (publications in the DBLP or projects in Codeplex) of person A_i, then t_k is the term belonging to the person A_i (t_k *in* T_{A_i}). Thus, we define (t_k *in* T_{A_i}) as the number of occurrences of term t_k in the description of projects in the input text T_{A_i}. This number is then approximated by the number of occurrences of term t_k in the all project's description (t_k *in* T). The higher value, the less relevant term t_k becomes. In addition, the result is approximated by T_{A_i}, because there is an assumption that T_{A_i}, which has a high cardinality, lower the importance of the individual terms, while low cardinality indicates that the author has only one subject matter. We can define the *relevance of author's terms* as:

$$R(T_{A_i}, t_k) = \frac{(t_k \; in \; T_{A_i})}{(t_k \; in \; T) + |T_{A_i}| - (t_k \; in \; T_{A_i})}. \tag{2.2}$$

And in normalized form:

$$R_{Norm}(T_{A_i}, t_k) = \frac{R(T_{A_i}, t_k)}{MAX(R(T_{A_i}, t_1), \ldots, R(T_{A_i}, t_{|T_{A_i}|}))}. \tag{2.3}$$

Because we have defined the relation between the persons and we can express the relevance of person's terms, we can assign the best suitable co-participant to given term. We can demonstrate the usage of significance of each co-participant as well. Our reflections were inspired by associative memory, where one is able to better recall the event, which is associated with something significant (although it was already forgotten). For a given person, it is significant the term, which associates him the best co-participant in the selected area.

The method extension, including the person's co-participant as a context, is then constructed analogically. The context is calculated for a given person according to the formula 4.8. Afterwards, the persons are selected from the evaluated list of co-participant.

Let us define a threshold θ, which indicates the term limit relevance to the person. If $R_{Norm} < \theta$ then R_{Norm} is set to 0. And the ContextScore is calculated by the equation:

$$ContextScore(T_{A_i}, T_{A_j}, P_{A_i}, P_{A_j}, t_k) = R_{Norm}(T_{A_i}, t_k) R_{Norm}(T_{A_j}, t_k) Q(P_{A_i}, P_{A_j}) \tag{2.4}$$

The entire procedure is shown on the Figure 2.1.

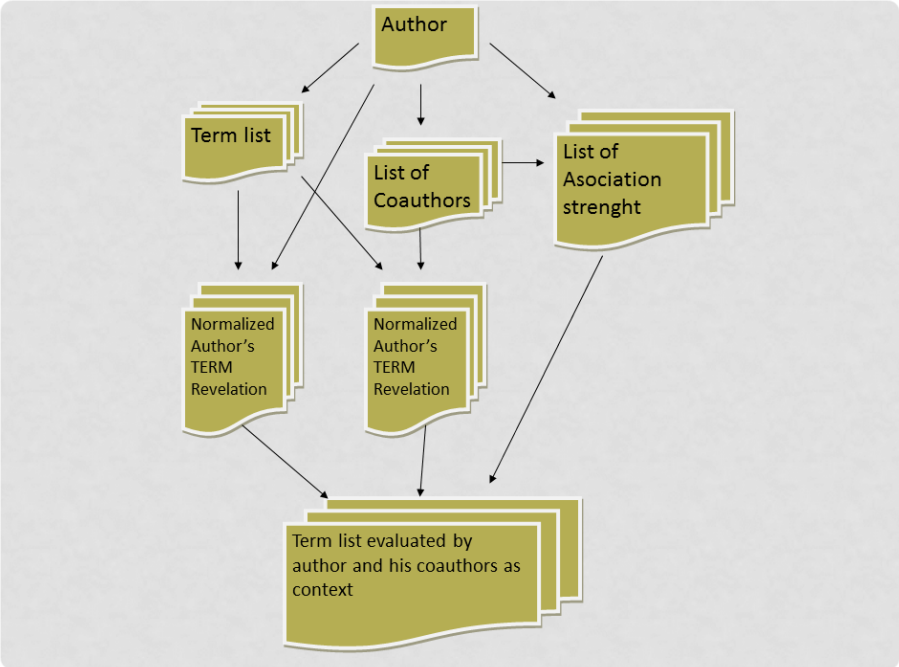

Figure 2.1: Procedure of using another author as context

CHAPTER III

Community Detection

The discovery and analysis of community structure in networks is a topic of considerable recent interest in sociology, physics, biology and other fields. Networks are very useful as a foundation for the mathematical representation of a variety of complex systems such as biological and social systems, the Internet, the world wide web, and many others [72, 31, 71]. A common feature of many networks is "community structure", the tendency for vertices to divide into groups, with dense connections within groups and only sparser connections between them [43, 74]. SNs [43] and information networks such as the web [39], have all been shown to possess strong community structure, a finding that has substantial practical implications for our understanding of the systems these networks represent. Newman and Girvan [77] proposed algorithms for finding and evaluating community structure in network. They used a "divisive" technique which iteratively removes edges from the network, thereby breaking it up in communities. The edges to be removed are identified by using one of a set of edge betweenness measures, of which the simplest is a generalization to edges of the standard shortest-path betweenness of Freeman. Than, their algorithms include a recalculation step in which betweenness scores are re-evaluated after the removal of every edge.

To detect communities, graph partitioning methods or hierarchical clustering has been applied. Originally, graph partitioning methods, based on edge removal [85], divide the vertices of a network into a given number of (non-overlapping) groups of a given size, while the number of edges between groups is minimal. Another technique currently in use is hierarchical clustering [92]. The idea behind it is to develop a measure of similarity $s(i, j)$ between pairs (v_i, v_j) of vertices, based on the network structure. Various beneficial metric functions have been proposed to help to solve the problems. Many community-finding algorithms are based on maximizing the quantity known as modularity [77, 21, 99] and any algorithm using modularity requires complete knowledge of the entire network.

Newman used eigenvectors of matrices for finding community structure in networks [75] and he used modularity in [76]. In [104] authors used a spectral clustering approach for community detection in graph. Their experimental results indicate that the new algorithms are efficient and effective at finding both good clusterings and the appropriate number of clusters. Eigenvectors of network complement reveal community structure more accurately [107]. Algorithms for finding community structure in very large networks can be found in [21]. Authors present a hierarchical agglomerative algorithm for detecting community structure. A spectral method for community detection is provided in [68].

In our research, our area of interest sometimes includes global structure networks and the communities found within them. We can use clustering algorithms or algorithms for graph partitioning to search for global communities (if the weight relations between objects are known). In the event we are not only interested in clearly distinguished groups, but also in the transitions between them (i.e. we are interested in individuals found in more than one community) we can use soft clustering algorithms (fuzzy C-mean, rough set clustering or our approach to spectral clustering).

3.1 Local community detection

Methods for searching local communities are often used in vast data collections. The aim of the algorithm for local community detection is to find a community — a user group -– based on local information that we have on individual users. Several local methods exist; all of them attempt to find the community containing a particular starting node [13, 4, 14, 20]. Typically, local community detecting techniques randomly start from a vertex v, and gradually merge neighbouring vertices one-at-a-time by optimizing a measured metric. In [21], Newman presents a hierarchical agglomerative algorithm for detecting community structure in large and sparse networks. In [86], authors propose an alternate local algorithm to detect communities, which outperforms the existing algorithms with respect to computational cost while maintaining the same level of reliability. An overview of methods for community detection is in [40].

In [9] author describes an evaluation comparing accuracy between five alternative, local community detection algorithms, in detecting two distinct types of community structures — vertex partitions that maximize modularity, and link clusters that maximize partition density in a variety of graphs.

3.2 Global community detection

Detecting communities is of great importance in sociology, biology and computer science, disciplines where systems are often represented as graphs. This problem is very complex and not yet satisfactorily solved, despite the huge effort of a large interdisciplinary community of scientists working on it over the past

few years [40]. For example Girvan and Newman have introduced a divisive algorithm where the selection of the edges to be cut is based on the value of their "edge betweenness" [77], a generalization of the centrality betweenness introduced by Freeman [41]. Consider the shortest paths between all pairs of nodes in a network. The betweenness of an edge is the number of these paths running through it. It is clear that when a graph is made of tightly bound clusters, loosely interconnected, all shortest paths between nodes in different clusters have to go through the few inter-clusters connections, which therefore have a large betweenness value. The single step of the Girvan and Newman detection algorithm consists in the computation of the edge betweenness for all edges in the graph and in the removal of those with the highest score. The iteration of this procedure leads to the splitting of the network into disconnected subgraphs that in their turn under go the same procedure, until the whole graph is divided in a set of isolated nodes. In this way the dendrogram is built, from the root to the leaves.

The spectral clustering is mostly used as a method for graph partitioning. We can use hierarchical (divisive) approach to spectral clustering and our algorithm detect overlap of communities (soft clustering).

3.3 Spectral Clustering

Spectral clustering has become one of the most popular modern clustering algorithms in recent years. It is one of the graph theoretical clustering techniques and is simple to implement, can be solved efficiently by standard linear algebra methods, and very often outperforms traditional clustering algorithms such as the k-means or single linkage (hierarchical clustering). A comprehensive introduction to the mathematics involved in spectral graph theory is the textbook of Chung [18] and we also recommend the survey of Schaeffer [92].

Spectral clustering algorithm uses the eigenvalues and eigenvectors of the Laplacian of the similarity matrix derived from the data set to find the clusters. In [36] Fiedler defines the second smallest eigenvalue $a(G)$ of the of Laplacian matrix $L(G)$ as algebraic connectivity of the graph G. In his honour, the corresponding eigenvector is called Fiedler vector. The other properties of the algebraic connectivity are in [37]. Donath and Hoffman [30] introduced the use of eigenvectors for the purpose of partitioning an undirected graph in a balanced way. The Spectral Partitioning Algorithm which uses Fiedler vector is summarized by Pothen et al. in [85].

Shi and Malik [60] treated image segmentation as a graph partitioning problem and propose a global criterion, the normalized cut, for segmenting the graph. They showed that an efficient computational technique based on a generalized eigenvalue problem can be used to optimize this criterion. In [28] Ding et al. proposed a new graph partition method based on the min-max clustering principle: the similarity between two subgraphs (cut set) is minimized, while the similarity within each subgraph (summation of similarity between all pairs of nodes within a subgraph) is maximized. The survey of data clustering relevant to the clustering document collection is in [51]. Von Luxburg et al. publicated a tutorial on spectral clustering in [98]. The analysis of spectral clustering is

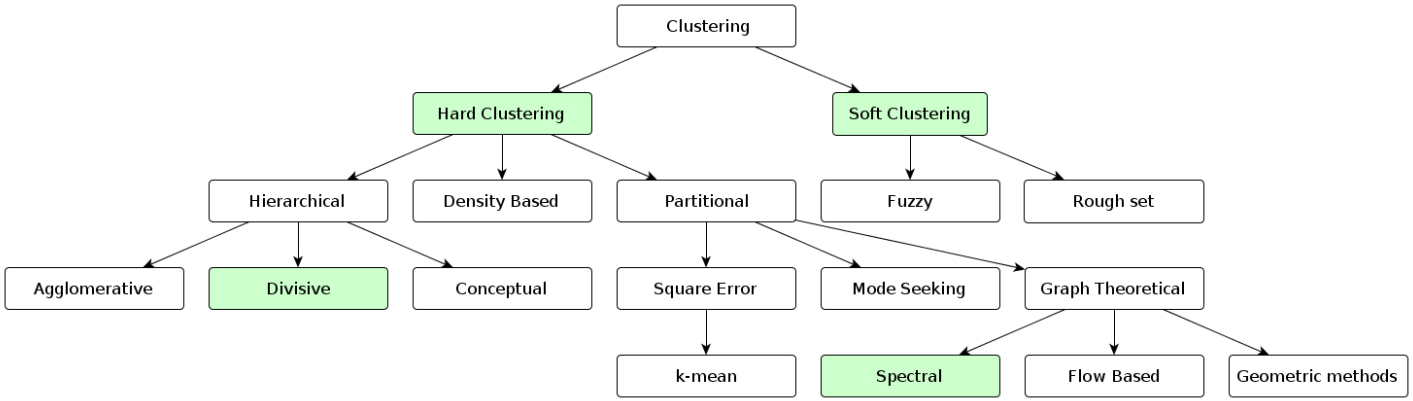

Figure 3.1: Spectral clustering inclusion in the taxonomy of Clustering

described in [52]. Kannan et al. developed a natural bicriteria measure for assessing the quality of a clustering. Cheng et al. [16] showed how to use the spectral algorithm studied in [52]. A practical implementation of the clustering algorithm is presented in [15]. Recursive spectral clustering algorithm is used in [24]. There Dasgupta et al. analysed the second eigenvector technique of spectral partitioning on the planted partition random graph model, by constructing a recursive algorithm. Spectral clustering approach to finding communities in graphs was applicated in [104].

3.3.1 Graph notation

Let $G = (V, E)$ be a *graph* with set of vertices $V = \{v_1, \ldots, v_n\}$ and the set E contains the edges of the graph. In an *undirected graph*, each edge is an unordered pair (u, v), where $u, v \in V$. In a directed graph, edges are ordered pairs. The *adjacency matrix* of the graph is the matrix $A = (a_{ij})$ $i, j = 1, \ldots, n$. If $(v_i, v_j) \notin E(G)$ then $a_{ij} = 0$, otherwise is $a_{ij} = 1$. For undirected graph it governs that A is symmetric. The *degree* of a vertex $v_i \in V$ is defined as $d_i = \sum_{j=1}^{n} a_{ij}$. The *degree matrix* D is defined as the diagonal matrix with the degrees $d_1, \ldots, d_n$ on the diagonal. The *degree distribution* for the graph G is $(k_0, k_1, \ldots, k_m)$, where $k_j =$ the number of vertices with degree j.

For a *weighted graph* G we have a weight function $w : E \to R$. It is for example function of the similarity between the nodes v_i and v_j. The weighted adjacency matrix of the graph is the matrix $W = (w_{ij})$ $i, j = 1, \ldots, n$. Then the degree of a vertex $v_i \in V$ in weighted graph is defined as

$$d_i = \sum_{j=1}^{n} w_{ij}.$$

The weighted degree matrix D is defined as the diagonal matrix with the weighted degrees $d_1, \ldots, d_n$ on the diagonal.

The *subgraph* $G_s = (V_s, E_s)$ is a graph all of whose vertices and edges are contained in a larger graph $G(V, E)$ ($V_s \subset V$ and $E_s \subset E$). A *path* from vertex v_i to vertex v_j is an ordered sequence $v_i, \ldots, v_j$ of distinct vertices in which each adjacent pair is linked by an edge. If each vertex in undirected graph G is reachable from each other vertex, then G is *connected*. A *component* of G is a maximal connected subgraph (ie a connected subgraph with vertex set V_s for which no larger set V_l containing V_s is connected). A *bipartite graph* (vertex set can be partitioned into 2 subsets, and there are no edges linking vertices in the same set)

Given a subset of vertices $S \subset V$ and its *complement* is denoted $\overline{S} = V \backslash S$. We consider two different ways of measuring the size of a subset $S \subset V$: $|S| =$ the number of vertices in S and $vol(S) = \sum_{v_i \in S} d_i$. It measures the size of S by the weights of edges originating from subset S.

Graph *cut*

$$cut(S, \overline{S}) = \sum_{v_i \in S, v_j \in \overline{S}} w_{ij}$$

is the sum of weight connections between two *clusters*.

3.3.2 Graph Laplacians and spectral bipartitioning

Graph Laplacian matrices are the main tools for spectral clustering. In this section we will defined different graph Laplacians and point out their important properties. We will distinguish between different variants of graph Laplacians. Note that in the literature, there is no unique convention which matrix exactly is called graph "Laplacian" and how the different matrices are denoted.

We shortly describe the minimum cut [105], ratio cut [46], Shi-Malik normalized cut [60] and min-max cut [28]. The minimum cut partition seeks to minimize the total link weights cut. The ratio cut measure is proportional to the total link weight cut, normalized by the sizes of the partitions. The Shi-Malik normalized cut measure is also a normalized measure, but the normalizing factor is the product of the total connectivity (valency) of the nodes in each partition.

3.3.2.1 The minimum cut

The optimal bipartitioning of a graph is the one that minimizes this cut value $cut(S,\overline{S}) = \sum_{v_i \in S, v_j \in \overline{S}} w_i j$. Wu and Leahy [105] proposed a clustering method based on this minimum cut criterion. However, as Wu and Leahy also noticed in their work, the minimum cut criteria favours cutting small sets of isolated nodes in the graph.

The objective function of the min-cut method is defined by:

$$\begin{aligned}
J_{MinCut} &= cut(S,\overline{S}) = \\
&= \sum_{v_i \in S, v_j \in \overline{S}} w_{ij} = \\
&= \frac{1}{4}\sum_{i,j=1}^{n} w_{ij}(q_i - q_j)^2 = \\
&= \frac{1}{4}(\sum_{i=1}^{n} d_i q_i^2 - 2\sum_{i,j=1}^{n} q_i q_j w_{ij} + \sum_{j=1}^{n} d_j q_j^2) = \\
&= \frac{1}{2}(\sum_{i=1}^{n} d_i q_i^2 - \sum_{i,j=1}^{n} q_i q_j w_{ij}) = \\
&= \frac{1}{2}(q^T D q - q^T W q) = \\
&= \frac{1}{2} q^T L q,
\end{aligned}$$

where $q \in R^n$ is the indicator vector of vertices belonging to clusters S and $\overline{S}$ such that $q_i = 1$ for $v_i \in S$ and $q_i = -1$ for $v_i \in \overline{S}$.

The solution minimizing the objective function will be equivalent to solve the following equation

$$(D - W)u = \lambda u$$

and the second smallest eigenvector of the unnormalized graph Laplacian matrix $L = D - W$ is related to a set that minimizes the cut. An overview of its properties can be found in Mohar [66, 67].

3.3.2.2 The ratio cut

Our goal is to solve the optimization problem

$$min_{S \in V} RatioCut(S, \overline{S}). \tag{3.1}$$

The objective function of the ratio cut method is defined by:

$$\begin{aligned}
J_{RatioCut} &= \frac{cut(S,\overline{S})}{|S|} + \frac{cut(S,\overline{S})}{|\overline{S}|} = \\
&= \frac{cut(S,\overline{S})}{|S|} + \frac{cut(S,\overline{S})}{|\overline{S}|} = \\
&= \sum_{v_i \in S, v_j \in \overline{S}} w_{ij}(\frac{1}{S} + \frac{1}{\overline{S}}) = \\
&= \frac{|V|}{|V|} \sum_{v_i \in S, v_j \in \overline{S}} w_{ij}(\frac{1}{S} + \frac{1}{\overline{S}}) = \\
&= \frac{\sum_{v_i \in S, v_j \in \overline{S}} w_{ij}}{|V|}(\frac{|S| + |\overline{S}|}{S} + \frac{|S| + |\overline{S}|}{\overline{S}}) = \\
&= \frac{\sum_{v_i \in S, v_j \in \overline{S}} w_{ij}}{|V|}(\frac{|\overline{S}|}{S} + \frac{|S|}{\overline{S}} + 2) = \\
&= \frac{1}{2|V|}\left[\sum_{v_i \in S, v_j \in \overline{S}} w_{ij}\left(\sqrt{\frac{|\overline{S}|}{|S|}} + \sqrt{\frac{|S|}{|\overline{S}|}}\right)^2 + \sum_{v_i \in \overline{S}, v_j \in S} w_{ij}\left(-\sqrt{\frac{|\overline{S}|}{|S|}} - \sqrt{\frac{|S|}{|\overline{S}|}}\right)^2\right] = \\
&= \frac{1}{2|V|}\sum_{i,j=1}^{n}(q_i - q_j)^2 = \\
&= \frac{1}{2|V|}q^T L q
\end{aligned}$$

where $q \in R^n$ is the indicator vector of vertices belonging to clusters S and $\overline{S}$ such that $q_i = \sqrt{|\overline{S}|/|S|}$ for $v_i \in S$ and $q_i = -\sqrt{|S|/|\overline{S}|}$ for $v_i \in \overline{S}$. Additionally, we have

$$\sum_{i,j=1}^{n} f_i = \sum_{i \in S}\sqrt{\frac{|\overline{S}|}{|S|}} - \sum_{i \in \overline{S}}\sqrt{\frac{|S|}{|\overline{S}|}} = |S|\sqrt{\frac{|\overline{S}|}{|S|}} - |\overline{S}|\sqrt{\frac{|S|}{|\overline{S}|}} = 0.$$

In other words, the vector q is orthogonal to the constant one vector $\vec{1}$. Finally, note that q satisfies

$$||q||^2 = \sum_{i=1}^{n} q_i^2 = |S|\frac{|\overline{S}|}{|S|} + |\overline{S}|\frac{|S|}{|\overline{S}|} = |\overline{S}| + |S| = n.$$

So the problem of minimizing objective function of ratio cut can be equivalently rewritten as

$$min_{S \subset V} q^T L q \text{ subject to } q \perp \vec{1}, ||q|| = \sqrt{n}.$$

This is an NP-hard discrete optimization problem as the entries of the solution vector q are only allowed to take two particular values. The obvious

relaxation in this setting is to discard the condition on the discrete values for q_i and instead allow $q_i \in R$. This leads to the relaxed optimization problem

$$min_{q \in R^n} q^T L q \text{ subject to } q \perp \vec{1}, ||q|| = \sqrt{n}.$$

By the Rayleigh-Ritz theorem it can be seen immediately that the solution of this problem will lead to the generalized eigensystem

$$(D - W)u = \lambda D u.$$

This approach introduces the size of clusters leading to balanced clusters. But in clustering concept a more important term is missing, the within cluster connections.

3.3.2.3 The normalized cut and the min-max cut

A successful method should take into consideration both inter and intra cluster connections and this was made in normalized cut [60] and min-max [28] cut methods. This methods have two constraints. The 1st constraint is a minimization of inter-connections (minimum of $cut(S, \overline{S})$) and the 2st constraint is a maximization of intra-connections (maximum of $cut(S, S)$ and $cut(\overline{S}, \overline{S})$). The objective function of the normalized cut method is defined by:

$$J_{NormalizedCut} = \frac{cut(S, \overline{S})}{vol(S)} + \frac{cut(S, \overline{S})}{vol(\overline{S})}$$

The objective function of the min-max cut method is defined by:

$$J_{MinMaxCut} = \frac{cut(S, \overline{S})}{cut(S, S)} + \frac{cut(S, \overline{S})}{cut(\overline{S}, \overline{S})}.$$

The normalized graph Laplacian matrix is defined as

$$L_{sym} = D^{-\frac{1}{2}} L D^{-\frac{1}{2}} = I - D^{-\frac{1}{2}} W D^{-\frac{1}{2}}.$$

3.3.3 Spectral ordering

Ding and He in [27] showed that a linear ordering based on a distance sensitive objective has a continuous solution which is the eigenvector of the Laplacian. Their solution demonstrate close relationship between clustering and ordering. They proposed direct K-way cluster assignment method which transforms the problem to linearisation the clustering assignment problem. The linearised assignment algorithm depends crucially on an algorithm for ordering objects based on pairwise similarity metric. The ordering is such that adjacent objects are similar while objects far away along the ordering are dissimilar. They showed that for such an ordering objective function the inverse index permutation has a continuous (relaxed) solution which is the eigenvector of the Laplacian of the similarity matrix.

The objective function of the spectral ordering method is defined by:

$$J_{SOrdering}(\pi) = \sum_{l=1}^{n-1} l^2 J_l(\pi) = \sum_{l=1}^{n-1} l^2 \sum_{i=1}^{n-l} w_{\pi_i, \pi_{i+l}}, \tag{3.2}$$

where the ordering is defined by the index permutation $\pi(1,2,\ldots,n) = (\pi_1,\ldots,\pi_n)$, the permuted similarity matrix is $(\pi W \pi^T)_{ij} = w_{\pi_i,\pi_j}$ and l is the fixed distance on the permuted order.

The goal is minimizing global ordering objective function:

$$min_\pi J_{SOrdering}(\pi).$$

Let us compute the optimal π. Let $l = |i-j|$ is distance between i and j. Than we can $J(\pi)$ rewrite as

$$\begin{aligned} J(\pi) &= \frac{1}{2}\sum_{i,j}(i-j)^2 w_{\pi_i,\pi_j} = \\ &= \frac{1}{2}\sum_{\pi_i,\pi_j}(i-j)^2 w_{\pi_i,\pi_j}. \end{aligned}$$

We can replace π_i by i in summation, because index i is permuted to π_i^{-1}, where π_i^{-1} is the inverse permutation and than we obtain

$$\begin{aligned} J(\pi) &= \frac{1}{2}\sum_{i,j}(\pi_i^{-1} - \pi_j^{-1})^2 w_{i,j} = \\ &= \frac{n^2}{8}\sum_{i,j}(\frac{\pi_i^{-1} - (n+1)/2}{n/2} - \frac{\pi_j^{-1} - (n+1)/2}{n/2})^2 w_{i,j}. \end{aligned}$$

For simplicity, we can shifted inverse index permutation

$$q_i = \frac{\pi_i^{-1} - (n+1)/2}{n/2} \in \{\frac{1-n}{n}, \frac{3-n}{n}, \ldots \frac{n-n}{n}\} \tag{3.3}$$

which satisfies

$$\sum_i q_i = 0, \sum_i q_i^2 = 1, \tag{3.4}$$

where $\mathbf{q}$ is scaled by $q_i \to (n^3/12 - n/3)^{-1/2} q_i$ which does not change the permutation. We can overwrite:

$$\sum_{ij}(q_i - q_j)^2 w_{ij} = \sum_{ij}(q_i^2 + q_j^2 - 2q_i q_j) w_{ij} = 2\mathbf{q}^T(D-W)\mathbf{q},$$

where D is diagonal matrix with $d_{ii} = \sum_j w_{ij}$. Therefore, we need to minimize $\mathbf{q}^T(D-W)\mathbf{q}$ for q_i taking those discrete values of Eq.3.3, subject to the constraints in Eq.3.4. Using a Lagrangian multiplier for the second constraint in Eq.3.4, minimization of $J_{SOrdering}$ becomes

$$min_q J_A, J_A = \frac{\mathbf{q}^T(D-W)\mathbf{q}}{\mathbf{q}^T\mathbf{q}}. \tag{3.5}$$

Finding the optimal solution for the discrete values of $\mathbf{q}$ is a combinatorial optimization problem, and is likely to have no polynomial-time optimal algorithms. However a continuous solution for $\mathbf{q}$ can be computed. We relax the restriction that q_i must take discrete values of Eq.3.3 in $< -1, 1 >$. With this, J_A

can be minimized by solving an eigenvalue problem. It is known that $\mathbf{q}$ is an eigenvector of the equation

$$(D - W)\mathbf{q} = \xi\mathbf{q}. \tag{3.6}$$

It is obvious that $\mathbf{q_0} = \mathbf{1} = (1, \ldots, 1)^T$ is an eigenvector with $\xi_0 = 0$. All other eigenvector are orthogonal to $\mathbf{q_0}$, i.e. the first constraint in Eq.3.4 is also satisfied. Therefore $\mathbf{q_1}$ is the desired continuous solution of the distance sensitive ordering.

Authors in [27] made a modification on the above solution which makes a direct connection to the scaled PCA. They used the new constraints:

$$\sum_i q_i d_i = 0, \sum_i q_i^2 d_i = 1, \tag{3.7}$$

where d_i is *volume* of vertex v_i (see section 3.3.1). With these constraints, the minimization problem of of $J_{SOrdering}$ becomes

$$min_q J_S, J_A = \frac{\mathbf{q}^T(D - W)\mathbf{q}}{\mathbf{q}^T D\mathbf{q}}. \tag{3.8}$$

Relaxing q_i to continuous values in $< -1, 1 >$, the solution for $\mathbf{q}$ satisfies the eigenvalue equation

$$(D - W)\mathbf{q} = \xi D\mathbf{q}. \tag{3.9}$$

Let $\mathbf{q} = D^{-1/2}\mathbf{z}$. Substituting it into Eq. 3.9, we obtain a eigenvalue equation

$$D^{-1/2}WD^{-1/2}\mathbf{z} = \lambda\mathbf{z}, \lambda = 1 - \xi. \tag{3.10}$$

3.3.4 Algorithm for spectral bisection

We used in our previous works [79, 78] algorithm for spectral bisection. We used Laplacian of similarity matrix and calculated Fiedler vector for finding spectral ordering.

3.3.5 Modularity - Quality of detected communities

To quantify the quality of the subdivisions we can use the modularity. Consider a particular division of a network into k communities. Let us define a $k \times k$ symmetric matrix $\mathbf{e}$ whose element e_{ij} is the fraction of all edges in the network that link vertices in community i to vertices in community j [77]. Networks with high modularity have dense connections between the nodes within community but sparse connections between nodes in different communities. Modularity is often used in optimisation methods for detecting community structure in networks [76]. The value of the modularity lies in the range $\langle -1/2, 1\rangle$. It is positive if the number of edges within groups exceeds the number expected on the basis of chance.

In terms of the edge weights, modularity $Q(C_1, \ldots, C_k)$ is defined over a specific clustering into k known clusters $C_1, \ldots, C_k$ as

$$Q(C_1, \ldots, C_k) = \sum_{i=1}^{k}(e_{ii} - \sum_{j=1, i\neq j}^{k} e_{ij})$$

where $e_{ij} = \sum_{(u,v)\in E, u\in C_i, v\in C_j} w(u, v)$ with each edge $(u, v) \in E$ included at most once in the computation.

Algorithm 3.1 Spectral bisection

1. Find all connected components in the graph.
2. For every connected components: create Laplacian matrix of component $L = D - W$, where W is the adjacency matrix with weights and with zero on the diagonal, D is the diagonal matrix with $d_{ii} = \sum_j w_{ij}$.
3. Find the eigenvector corresponding to second smallest eigenvalue of L.
4. Divide the component based on the sorted eigenvector.
   - Split eigenvector in 0.
   - Split eigenvector in mean or median.
   - Split eigenvector in gap.
5. Recursion on the obtained components (back to the step 2).

3.4 Left-Right algorithm for community detection

Upon completing our study of various modifications of algorithms for spectral clustering, we designed our own algorithm Left-Right-Oscillate (LRO) for detecting communities within complex networks. This algorithm utilizes spectral ordering where similar vertices are closer to indexes and less similar vertices are further from indexes. When determining the ordering, it is necessary to calculate the eigenvector of the second smallest eigenvalue of the matrix $L = D - W$. Since we have designed our algorithm for large amounts of data in a complex network, we used Lanczos method to calculate the Fiedler vector. Once the Fiedler vector was calculated, we detected appropriate gaps that divide the vertices of a graph into communities. As observed in the experiment (see Figure 3.2), this type of separation into gaps leads to several badly-assigned subgraphs. This is due to the fact that the Fiedler vector is only linear ordered, as is revealed in our data collection.

The Left-Right method we have designed for incorporating small subgraphs into larger communities gradually increases modularity in a given calculation (see Figure 3.3).

Spectral ordering minimizes the sum of weighted edges multiplied to the power of the difference in index nodes with edge incidence. The calculation used for this equation is the given eigenvector of the second smallest eigenvalue (Fiedler vector) matrix $L = D - W$. A visualized Fiedler vector (see figure 3.4) and ordered matrix similarity (in agreement with the Fiedler vector) reveals the creation of several natural clusters which is assigned by our algorithm (see Figure 3.5).

For finding the Fiedler vector of Laplacian above a large, sparse and symmetric matrix representative of the evaluated network, we used Lanczos method to partially solve the eigenvalue problem. To determine the dimension of Krylov subspaces (for a more precisely calculated Lanczos method) we used modularity

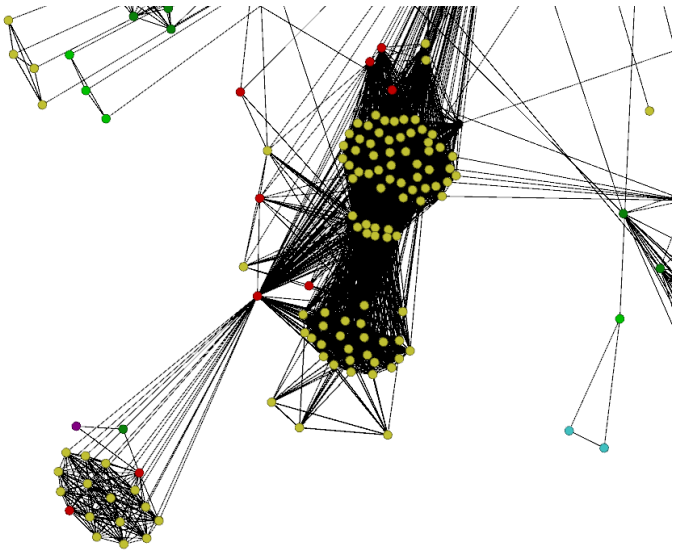

Figure 3.2: Some communities have different colors of vertices - some vertices are badly-assigned.

for determining the quality of a detected community. In [27], there is an example of a symmetric Laplacian $L_{sym} = D^{-\frac{1}{2}}LD^{-\frac{1}{2}}$. Our experiment revealed that this solution is only appropriate for some networks.

The next step for the algorithm is to order indexes of vertices $v_i \in V$ for all $i = 1, \ldots n$ in compliance with ordering using Fiedler vector values. Because we want to find communities that are easily detected in a visual representation when ordered by a similarity matrix, we must determine where one community in a linear order ends and the next begins (find two nodes that belong to various communities). For this reason, we have calculated the value of antidiagonal sums above an ordered set of vertices that capture a cluster overlap in neighbouring vertices v_i. We define cluster crossing as the sum of a small fraction of the pairwise similarities. This is aided by linear ordering data points. The goal is to find nodes that lay in areas with fewer edges. These vertices lie close to locales with minimum function that are attached by approximation of a cluster overlap discrete function Sum_i (see section 3.3.3) (see figure 3.4). We assigned this approximation using the spline function, allowing for easy calculations of both the first and second derivation, which are used to assign local extremes. Between the two local maximum extremes of this function, there lie two vertices. In this area, these vertices represent a maximum gap (the difference in their Fiedler vector value). This gap determines the border between two potential communities.

Using this method, we have found the natural amount of "communities" above a given evaluated network (see figure 3.5). Since the precision with which the Fiedler vector is calculated is a determining factor (see figure 3.6) and since

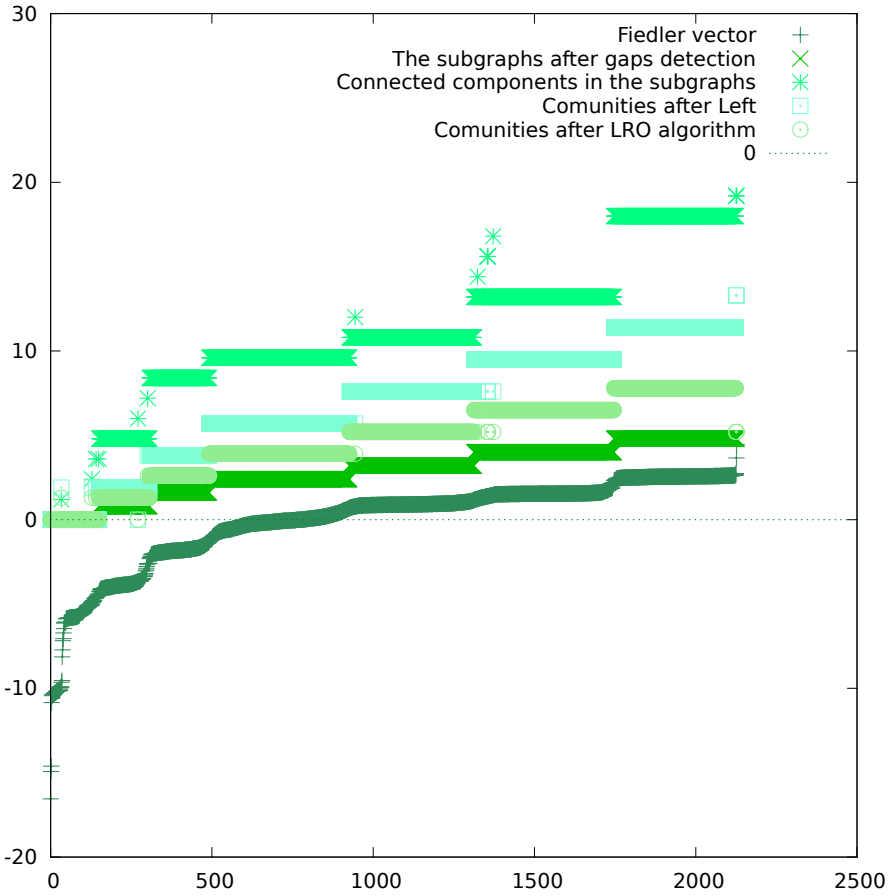

Figure 3.3: The algorithm's evolution – gaps determine subgraphs – connected components in every subgraph – communities after Left part of the LRO algorithm – communities after LRO.

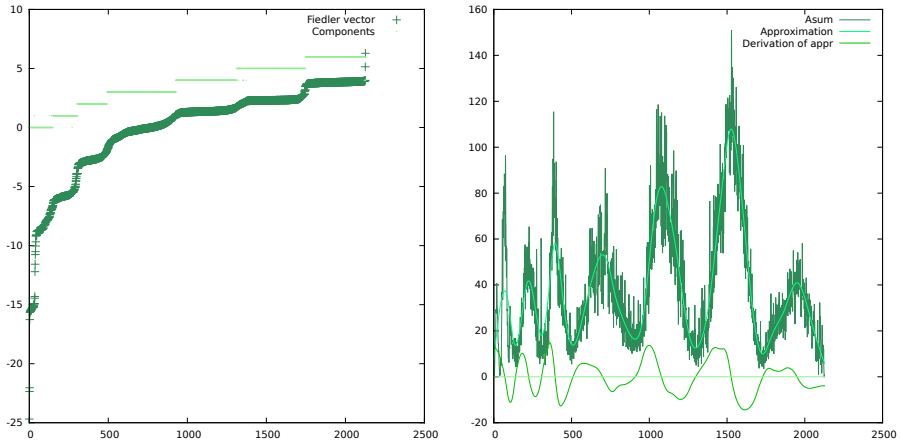

Figure 3.4: Fiedler vector and obtained components; function $Asum_i$ with the spline approximation and first derivation of the approximation

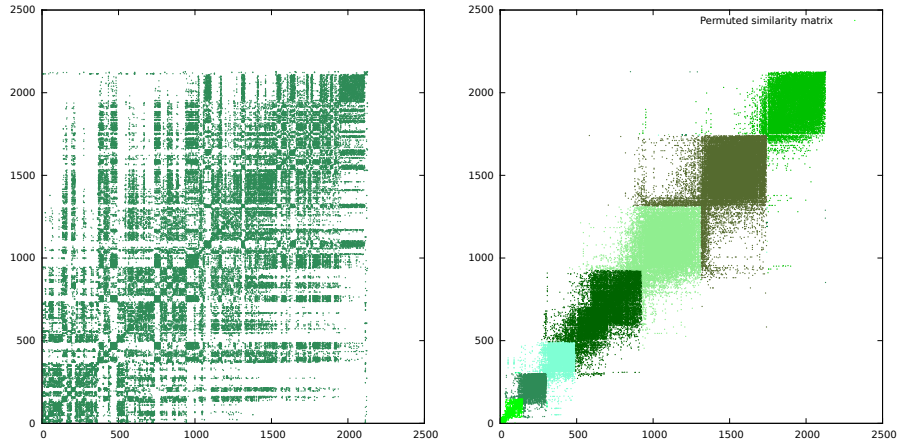

Figure 3.5: Similarity matrix and permuted similarity matrix (natural number of communities is 7)

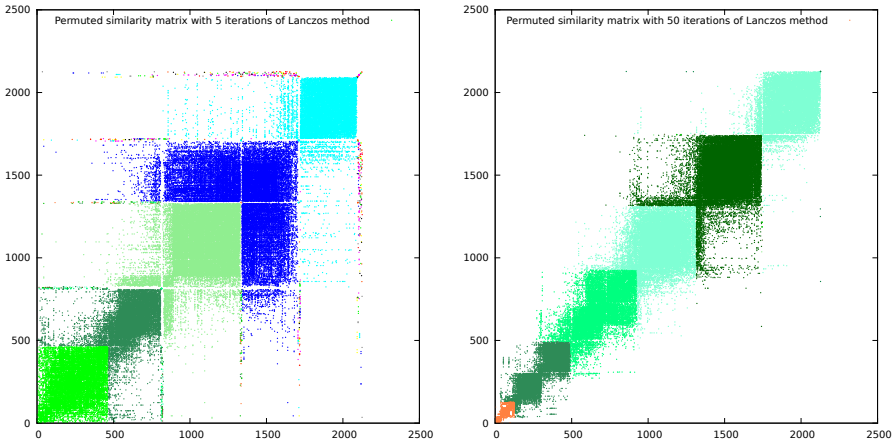

Figure 3.6: Permuted similarity matrix with 5 and 50 iterations in Lanczos method

in some cases vertices are incorrectly assigned, the result is an irrelevant component.

The benefit of using our algorithm lies within its ability to assign isolated nodes (or very small subgraphs with selected sizes) to the nearest, most suitable, connected component that creates the nucleus of a future community. Within our assignments, we gradually arrive at a set of vertices V separated by gaps in individual subsets V_k. If the found set V_k does not create a connected subgraph ($G_k = (V_k, E)$), we determine all connected components in this subgraph. The maximum connected subgraph then creates the nucleus of this community and all subgraphs smaller than the selected size are moved to the right. We then attempt to reassign the subgraph to the next subset of vertices V_{k+1}. Due to the linear nature of spectral ordering, it is presumable that subgraphs not yet assigned are reordered to the next subset of vertices. This means that we add the vertices of these subgraphs to the vertices of the next subset (that came into existence along gaps and creates a subgraph of the original graph with a set of vertices V_{k+1}). Then we test the connectivity of subgraph G_{k+1}, which was expanded by nodes from the previous, unassigned subgraph. We go through the entire, spectrally ordered set of graph vertices employing this method. At the end of this process, we have created the most relevant of components within which we assign small subgraphs that are not yet assigned. We then repeat this approach in the opposite direction – going from right to left – and we try to add vertices for inspection in a subgraph. We may then assign the vertices to a connected subgraph with adjacency to a vertex of a given subgraph (see Figure 3.7).

Once we assign a subset of vertices using gaps, and once we have detected connected components from left to right and vice versa, we always calculate the modularity for the obtained separation of graphs into subgraphs. Our results have revealed that our Left-Right method increases modularity (see table 3.1). The resulting connected subgraphs then create the structure of communities in the graph.

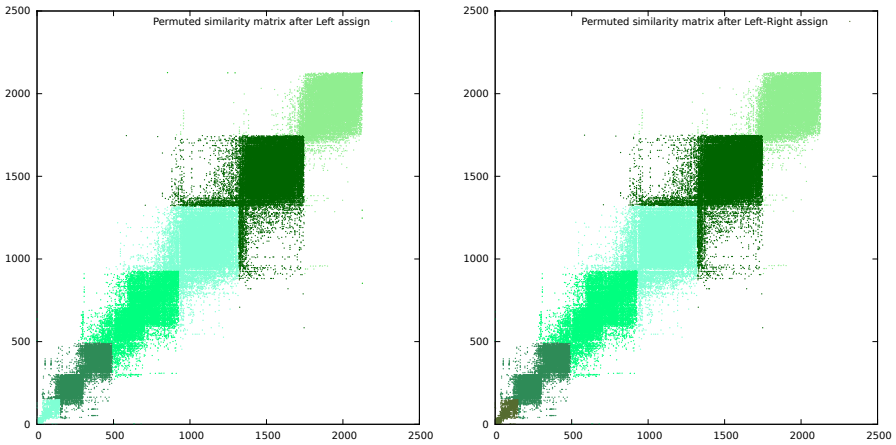

Figure 3.7: Permuted similarity matrix after Left and after Left-Right part of algorithm Left-Right

	before LRO	components	after LRO	components
Laplacian	0.27	11	0.36	4
normalized-cut	0.342	7	0.348	3

Table 3.1: Modularity before and after Left-Right-Oscillate Algorithm (with size of smallest community = 5) for Zachary karate club (we use Laplacian $L = D - W$ or for normalized-cut $L_S = D^{-1/2}WD^{-1/2}$.

Algorithm 3.2 Left-Right algorithm for community detection

Input: similarity matrix $W = w_{i,j}$ for $i = 1, \dots n$ and S_c size of smallest communities.
Output: communities C_k, modularity of detected communities.

1. We create Laplacian $L = D - W$ using a matrix of similarity W of a connected graph $G = (V, E, W)$.
2. We calculate the Fiedler vector (the second eigenvector of Laplacian).
3. We reorder vertices according to Fiedler vector.
4. We calculate the sums of antidiagonals $Asum_i = \sum_{\forall j} w_{i-j,i+j}$ a $Asum_{(i\pm1/2)} = \sum_{\forall j} w_{i-j,i+j\pm1}$for all $i = 1, \dots n$ and calculate $sum_i = Asum(i-1/2)/4 + Asum(i)/2 + Asum(i+1/2)/4$.
5. We approximate the discrete function sum_i by its spline and we determine its first and second derivation. We then find all local minimums and maximums.
6. We assign maximum gaps that lie between two local maximums. We divide the set of vertices according to its gaps. We obtain subsets $SS_k \subset V$, where $k = 1, \dots K$ is the amount of subsets.
7. Using the Left-Right assigning algorithm, we detect a community.

Algorithm 3.3 Left-Right assigned

Input: subsets $SS_k \subset V$, where $k = 1, \dots K$, S_c size of smallest communities.
Output: communities C_k, modularity of detected partitioning.

1. For every subset $SS_k \subset V$ $k = 1, \dots K$ we find connected components C_j, which are greater than the selected size $|C_j| \geq S_c$. These components create communities. We add the rest of the vertices $v_i \in SS_k - \bigcup V(C_j)$ to the next subset of vertices SS_{k+1}. We continue repeating this method by step 1 until we reach the end of ordered vertices.

2. Once we go through all subsets of nodes, connected components are assigned to C_j for $j = 1, \dots J-1$ and C_J contain a set of connected components smaller than the selected size.

3. We employ the same approach going right-left. We begin with subset of vertices $V(C_{J-1}) = V(C_{J-1}) \cup V(C_J)$.

We improved the Algorithm Left-Right assigned so, that we try to assign in the each step rest of the vertices to the previous or next subset of vertices. Than the Algorithm Left-Right-Oscillate (LRO) has increased modularity.

Algorithm 3.4 Left-Right-Oscillate assigned

1. For subset $SS_1 \subset V$ we find connected components C_j, which are greater than the selected size $|C_j| \geq S_c$. These components create communities. We add the rest of the vertices $v_i \in SS_1 - \bigcup V(C_j)$ to the next subset of vertices SS_2.

2. For every subset $SS_k \subset V$ $k = 2, \dots K$ we find next connected components C_j, which are greater than the selected size $|C_j| \geq S_c$. These components create communities. We attempt to assign other vertices to the previous community, which was established in the previous step. If the vertex has no edge leading to the previous community than we add the vertex to the next subset of vertices $SS_{k+1} \subset V$. We continue repeating this method by step 2 until we reach the end of ordered vertices.

3. Once we go through all subsets of vertices, connected components are assigned to C_j for $j = 1, \dots J-1$ and C_J contain a set of connected components smaller than the selected size.

4. We employ the same approach going right-left without "oscillation". We begin with $V(C_{J-1}) = V(C_{J-1}) \cup V(C_J)$.

We can use our LRO algorithm in the hierarchical way for the next improvement. This approach is usable for a very large network where the structure is very complex and the one level of community detection is not enough.

Algorithm 3.5 Hierarchical Left-Right-Oscillate algorithm

Input: similarity matrices $W_k = w_{i,j}$ for $i = 1, \dots n$ for all connected components, S_c size of smallest communities, $minM$ minimal modularity.
Output: communities C_k, modularity of detected communities

1. We determine via our LRO algorithm all communities for all connected components .

2. We create new connected components from all communities which are greater then S_c and their modularity is greater the $minM$ so, that we cancel edges between communities and then we continue by step 1.

CHAPTER IV

Experiments

4.1 Discovering of social network

In this part of the doctoral thesis we discuss about discovering, deriving and evaluating of relations in the specified network. Our study was concentrated on the scholar information system – Edison, the learning management system – Moodle, the Microsoft's open source project hosting web site – CodePlex and the computer science bibliography website – DBLP.

4.1.1 Edison - scholar information system

We have used data collection from the university information system Edison used for organization, planning and evidence of students' evaluation at FEECS, VŠB - Technical University in Ostrava, Czech Republic. We have selected three consecutive academic years from 2008 to 2011, for which we have explored the behaviour of students during their studies. Our motivation was to find, whether exist groups of students, which repeatedly attend the same courses and the tutorials. The main hypothesis was, that the students attending the same (or sufficiently similar) courses and their tutorials together (from various reasons), could have similar characteristics or could know each other. From the view of social network analysis, we can say that between such students than exist relations on the basis of their similar behaviour. Our research was then oriented to finding the reasons of such behaviour, and to discovering of potential influence to their study results.

The data collection consisted of 85 161 records with information about students and academic year, semester (winter, summer), name of course and study group (concrete tutorial of course; one course can consist of several tutorials held simultaneously for multiple study groups due to the organizational reasons). Furthermore, for further research purposes, we have obtained more information

about the students like start year of student's studies, study program, previous graduated school and place of this school.

We have constructed bipartite graph from set of students and set of tutorials $S \times T$ with 85 161 edges. In this bipartite graph the edge have represented the situation, that given student s_i has attended the tutorial t_l.

From this bipartite graph we have constructed matrix of student $\times$ student, where for each pair of students s_i, s_j was defined number of common tutorials t_{ij} during the six consecutive semesters. From this matrix was constructed weighted graph of students, while the edge weight was defined by the following Eq. 4.1:

$$w(s_i, s_j) = \sum_{k=1}^{5} \sqrt{t_{ij}^k t_{ij}^{k+1}}, \tag{4.1}$$

where k is number of semesters. Because of the different length of students' studies, each weight was divided by the given number of transitions between the semesters, which students s_i a s_j have attended together.

The obtained graph consisted of 2 585 nodes and 319 482 edges; 268 197 edges have the weight equal to 0. After their reduction we had obtained 2 133 nodes.

4.1.2 Discovering of network in LMS Moodle

The analysed data collections are stored in the Learning Management System (LMS) Moodle logs used to support eLearning education at Silesian University, Czech Republic.

The logs consist of records of all events performed by Moodle's users such as communication in forums and chats, reading study materials or blogs, taking tests or quizzes etc. The users of this system (students, tutors, and administrators) are members of a community which aims to provide the appropriate services and guidance to its members, to make them achieve their objectives successfully. The authors are interested in studying students' activities in the Moodle system and in discovering the latent social network created from groups of students with similar patterns of behaviour.

Data anonymization was implemented during the data preprocessing phase, and the study was limited to investigating the events performed only by students. Let us define a set of students $s \in S$, set of courses $c \in C$ and term *Event* as a combination of Event prefix $p \in P$ (e.g. course view, resource view, blog view, quiz attempt) and a course c. An event then represents an action performed by student $s \in S$ in certain course c in LMS. On the basis of this definition, we have obtained *Set of Events* $e_i \in E$, which is represented by pairs $e_i = (p_j, c_k), j \in \{1, \ldots |P|\}, k \in \{1, \ldots |C|\}$ ordered by TimeStamp.

After that we obtain set of activities $a_j \in A$. *Activity* is a sequence of events $a_j = \langle e_1, e_2, \ldots, e_n \rangle$, performed by certain student s in a certain course c during the optimal time period. In our previous experiments we found the 30 minutes time period to be the most effective time interval. The findings showed that in shorter time periods (5 minutes) students were performing only non-study activities, and in longer periods there was not a significant activity difference (that means activities were very similar). For detailed information

see our previous work [32]. Similar conclusion was presented by Zorrilla et al. in [110].

Two matrices were obtained to represent the data: a Student matrix T ($|S| \times |A|$), where a row $(t_1, t_2, \ldots, t_{|A|})$ represents a subset of activities performed by the student in the Moodle system, and a matrix of similarity P ($|S| \times |S|$), which is derived from matrix T, and defines students' relationships using their similar activities. The similarity between two students (vectors) was defined by cosine measure [90].

$$p_{i,j} = \frac{\sum_{k=1}^{n} t_{ik} t_{jk}}{\sqrt{\sum_{k=1}^{n} t_{ik}^2} \sqrt{\sum_{k=1}^{n} t_{jk}^2}} \tag{4.2}$$

Matrices T and P are very large and sparse because of the large number of activities performed by students. Therefore, the visualization of the latent ties between students with similar behaviour was very hard and unfeasible. One of our goals was the reduction of that high number of activities using specification of smaller groups with characteristic activities.

4.1.3 CodePlex – network of developers

CodePlex is open source project hosting web site from Microsoft. CodePlex can be used to find open source software or create new projects to share with the world. Codeplex.com is 11 years old; it is ranked 2.107 in the world, a low rank means that this website gets lots of visitors. Its primary traffic goes from United States and is ranked 3.175 in United States. It has 104 subdomains with traffic. It has 136,500 visitors per day, and has 436,800 page views per day. CodePlex is mainly used by developers for collaboration on projects, sharing source codes, communication and software development. Generally, registered users can participate in multiple projects, discussions, adding the source code and documentation, issue a release, etc. Some of the users have defined a specific role within the project for which they work. Each user has his own page, where he can share information about himself, his projects on which he currently works, and the most recent activities. The CodePlex projects themselves can be considered as a very interesting source of information. In addition to the list of users and roles, CodePlex enables to register keywords, to add a description of the project, number of visits, a status, a date of creation, url and other information about the project. All activities are carried out on CodePlex by a particular user within a specific project.

Database which was created as a result of data obtained from CodePlex.com, consists of 6 main tables: User, Project, Discussions, RecentActivity, Membership and SourceCode (see Table 4.1).

In CodePlex, we can see two types of entities: users and projects. Both are represented by tables that contain specific characteristics. The table User contains informations about the users such as login, personalStatement, createdOn, lastVisit and url of user page. The table Project contains some characteristics of project in Codeplex: tags, date of created on, status, license, pageViews, count of visits, description and url of project page.

The undirect connection between the user and the project is implemented through activities within the scope of the project. These activities are in the

Table	Number of records
User	96251
Project	21184
Discussion	397329
RecentActivity	72285
Membership	126759
SourceCode	610917

Table 4.1: The CodePlex database tables

Activity	Meaning
SourceCode	records about added projects
Discussion	discussions about the project and the responses of individual users
RecentActivity	check-ins, task records, add Wiki information, notes about Release version etc
Membership	able to trace the users' participation in the projects and their assigned role

Table 4.2: The CodePlex activities

database CodePlex divided into different types: SourceCode, Discussion, RecentActivity and Membership (see Table 4.2).

We can represent CodePlex as a bipartite graph of users and projects, where the edge between the user and the project is defined as a user's activity in a project.

If we look at the data that we have in the table User, we are not able to define the user's profile. It consists of the field of interest, what he deals with, the programming language he uses and at what level. PersonalStatement attribute is used to describe the user. However, from the total set of our users downloaded, there was not a single one, who would fill it up. On the other hand, the project has enough information defined – which fields are concerned, how long it lasted, whether it is completed, which technology it is used, etc.

The main attribute, carrying the largest set of information, is the project Description – the description of the project itself.

Using activities such as user links to the projects, we are able to determine with some probability an area of specialization and a work of each user. For example, if a user is working on three projects written in .NET and one in Java, we could include him in .NET programmers with high probability, and less likely recommend him as a Java programmer.

In other words, terms or description of the project may not only help us to provide more information about projects, but also to determine the user's area of interests or abilities. As a result, the way we are able to compare user attributes determines the similarity to other network participants.

4.1.3.1 Graph of Collaborators

Whenever we think about collaboration between two persons, we not only look at the relationship itself, but also at the context. It is clear that depending on context, the strength of relationship changes. Therefore, we divide collaboration into two main parts *Developers' Relationship* and *Relationship Context*. We consider the relation between developers and the terms that describe the context between developers.

Moreover, the developers have additional attributes. Usually it could be publications, teams, organizations, projects, etc. We called it attribute domain; in our case D_{CP_d} is a set of projects in Codeplex, then CP_D are attributes for all D_i developers, where the objects are one developer's attributes described as $CP_{D_i} \subseteq D_{CP_D}$.

4.1.3.2 Developers' and Context Relationship

We describe a developers' relationship as commutative operation on Cartesian product of developer's attribute $X \times X$, where output is mapped to the set of real numbers $\mathbb{R}$.

We use Jaccard coefficient ([26] for evaluation of developers' relations using their attributes.

$$Attribute_{Score}(CP_{D_i}, CP_{D_j}) = \frac{|CP_{D_i} \cap CP_{D_j}|}{|CP_{D_i} \cup CP_{D_j}|} \tag{4.3}$$

As we discussed above, every developer has it's attributes. Moreover, each project has a description text. If we use lexical analysis on this text, we can define a term set for every developer as T_{D_i} and this term set contains all terms of projects, which developer D_i participated. The extracted text is proceed to methods, which remove words that do not carry any important information. The main issue of this experiment was not to describe this kind of methods. More could be found in [84, 55].

Term set T consists of all developers term sets $\{T_{D_0}, T_{D_1}, \ldots, T_{D_n}\} = T$, when the domain for terms T could be obtained as union of all terms extracted for each person $D_T = T_{D_0} \cup T_{D_1} \cup \cdots \cup T_{D_n}$.

The whole process of obtaining term sets is described in [65], so we just reminding $(t_k \; in \; T_{D_i})$ stands for the number of terms t_k by T_{D_i} and $(t_k \; in \; T)$ stands for the number of terms t_k in descriptions of all projects by T.

We can evaluate association between the selected term $t_k \in D_T$ and a developer $D_i \in D$:

$$R(T_{D_i}, t_k) = \frac{(t_k \; in \; T_{D_i})}{(t_k \; in \; T) + |T_{D_i}| - (t_k \; in \; T_{D_i})} \tag{4.4}$$

We normalize $R(T_{D_i}, t_k)$ such that $R_{Norm}(T_{D_i}, t_k) \in < 0, 1 >$:

$$R_{Norm}(T_{D_i}, t_k) = \frac{R(T_{D_i}, t_k)}{MAX(R(T_{D_i}, t_1), \ldots, R(T_{D_i}, t_{|T_{D_i}|}))} \tag{4.5}$$

Evaluation of the whole relationship context of two persons D_i and D_j has two steps. First, we compute association between D_i and select term t_k, and between the second developer D_j and t_k separately. Afterwards, because each

part is already evaluated by real number, we combine both results in the same way; we can combine the whole result in one equation. In CodePlex we see the description text for the developer as the all description of all projects he is working on, joined together. We obtain equation for the $Context_{Score}$:

$$Context_{Score}(T_{D_i}, T_{D_j}, t_k) = R_{Norm}(T_{D_i}, t_k)\, R_{Norm}(T_{D_j}, t_k) \tag{4.6}$$

4.1.3.3 Collaboration – Whole Score

The last step is to define Score, which consists of $Attribute_{Score}$ and $Context_{Score}$:

$$Score(CP_{D_i}, CP_{D_j}, T_{D_i}, T_{D_j}, t_k) = Attribute_{Score}(CP_{D_i}, CP_{D_j}) Context_{Score}(T_{D_i}, T_{D_j}, t_k) \tag{4.7}$$

This equation evaluates the relation between developers depending on the selected words, which represent the context. So we get a evaluation for the new subnet, which is specified by the selected terms.

4.1.3.4 Construction of the Graph

To describe the network of collaboration, we use standard weighted graph $G(V, E)$, where a weighted function is defined as $w : E(G) \mapsto \mathbb{R}$, when $w(e) \geq 0$.

The determination of set D is simple, because objects of vertices set V match with objects of set D, so $V = D$. However, we can do the same with all the possible pairs from set D to assign a set of edges E; it is better to design the algorithm to each implementation at first, and to reduce the number of useless computations. In addition, we must choose term t_k for function w, which reflects the context. Because only the commutative operations are used, we do not need to take into consideration the order of attribute objects in function parameters. Moreover E is two-object set, where the order of objects does not matter, so the evaluating is done just once.

When we construct graph based on developers' projects relationship, we use $Attribute_{Score}(CP_{D_i}, CP_{D_j})$ as w, where no term is needed, then simply $V = D$, which means that every developer is a vertex in the graph. Then, for each developer $D_i \in D$ we find collaborators D_{i_C} and for each collaborator $D_j \in D_{i_C}$ we create two-object set $\{D_i, D_j\}$, which corresponds with an edge in the graph. Equation 4.3 is then used to evaluate the edge.

The function $Score(CO_{D_i}, CO_{D_j}, T_{D_i}, T_{D_j}, t_k)$ is used for evaluating the edges in the context of the term. The only difference is, that majority of developers has not chosen term in their description text, so the result will be 0 and no edge would exists. Hence, we first determine subset of developers $D_{t_k} \subseteq D$ for those that have a term in their description text, followed by the same steps described in the last paragraph to compute developers' projects relationship. Then, the term t_k is used for computation of the second part in $Context_{Score}(T_{D_i}, T_{D_j}, t_k)$. Finally, we calculate the whole $Score$ by multiplication of both parts.

4.1.4 DBLP – co-author network

Very interesting source of information about scientific publishing in computer science is database DBLP [1]. Digital libraries are collections of resources and services stored in digital formats and accessed by computers. Studying them offers an interesting case

[1] Online access to DBLP database is available on http://dblp.uni-trier.de/

study for researches for the following reasons: firstly, they grow quickly; secondly, they represent a multidisciplinary domain which has attracted researchers from a wide area of expertise. DBLP (Digital Bibliography & Library Project) is a computer science bibliography database hosted at University of Trier, in Germany. It was started at the end of 1993 and listed more than one million articles on computer science in January 2010. These articles were published in Journals such as VLDB, the IEEE and the ACM Transactions and Conference proceedings. Besides DBLP has been a credible resource for finding publications; its dataset has been widely investigated in a number of studies related to data mining and social networks to solve different tasks such as recommender systems, experts finding, name ambiguity, etc. Even though, DBLP dataset provides abundant information about author relationships, conferences,and scientific communities. However, it has a major limitation; its records provide only the paper title without the abstract and index terms.

4.1.4.1 Co-authoring in DBLP

Besides the computation of evaluated term set itself, we can compute *association strength* between the two authors. This method is not only interesting by itself, but it is also essential for extended evaluation of the term list by selected context. Co-authors' relevancy is based on the co-authoring on the same articles. This relevancy is then approximated by Jaccard coefficient [26].

Let A be a set of all authors in DBLP. We define then single author A_i, for who we want to evaluate the strength of association with the other co-authors. In DBLP, co-authoring could be computed in a way that we go all the author's articles in DBLP and note all the co-authors participated. Afterwards, we can order them by the frequency of the articles published together. The set of co-authors of author A_i is marked as C_{A_i}. Let set P be a set of the all publications in DBLP and P_{A_i} be a set of the all publications of author A_i.

The association strength between the authors A_i and A_j can be defined with Jaccard coefficient that reflects mainly the proximity of both authors from the number of their common publications:

$$Q(A_i, A_j) = \frac{|P_{A_i} \cap P_{A_j}|}{|P_{A_j}| + |P_{A_i}| - |P_{A_i} \cap P_{A_j}|} \tag{4.8}$$

If this method is applied to all the authors in DBLP, we obtain weighted undirected graph that can be considered as a synthetic co-authors network (with re-weighted edges between authors). More about this research area can be found in [29, 91].

4.1.4.2 Searching Relations between Authors on the Basis of Association

The keyword evaluation method was till the certain point inspired by the original article [69]. In this case, however, we cannot use the association strength, as in the calculation of the relevance between the two authors based of the articles.

If we define a set T as the set of all terms of the DBLP and T_{A_i} as the set of all the terms that could be found in the titles of articles by A_i, then t_k is the term belonging to the author A_i (t_k *in* T_{A_i}). Furthermore, we define (t_k *in* T_{A_i}) as the number of terms t_k in the headlines of articles by T_{A_i}. This number is then approximated by the number of term t_k in the titles of articles DBLP (t_k *in* T). The higher value, the term becomes t_k less relevant. In addition, the result is approximated by T_{A_i}, because there is an assumption that T_{A_i}, which has a high cardinality, lower the importance of the individual terms, while low cardinality indicates that the author has only one subject matter:

Number of vertices	2128
Number of edges	51279
Number of connected components	1
Size of biggest component	2128
Average path length	3.404
Diameter	9
Average Degree	48.195
Average Weighted Degree	75.891

Table 4.3: Characteristics of Edison

$$R(T_{A_i}, t_k) = \frac{(t_k \; in \; T_{A_i})}{(t_k \; in \; T) + |T_{A_i}| - (t_k \; in \; T_{A_i})} \tag{4.9}$$

$$R_{Norm}(T_{A_i}, t_k) = \frac{R(T_{A_i}, t_k)}{MAX(R(T_{A_i}, t_1), \ldots, R(T_{A_i}, t_{|T_{A_i}|}))} \tag{4.10}$$

4.1.4.3 Method of Using Another Author as Context

Because we have defined the relation between the authors and we can express the relevance of author's terms, we can assign to given term the best suitable co-author. We can demonstrate the usage of significance of each co-author as well. Our reflections were inspired by associative memory, where one is able to better recall the event, which is associated with something significant (although it was already forgotten). For a given author is significant the term, which associate him the best co-author in the selected area.

The method extension, including the author's co-author as context, is then constructed analogically. The context is either selected randomly, or is first calculated for a given author according to the equation 4.8. Afterwards, the authors are selected from the evaluated list of co-authors.

Let us define a threshold θ, which indicates the term limit relevance to the author. For lower value than threshold is the result of R_{Norm} equal 0:

$$ContextScore(T_{A_i}, T_{A_j}, P_{A_i}, P_{A_j}, t_k) = R_{Norm}(T_{A_i}, t_k) R_{Norm}(T_{A_j}, t_k) Q(P_{A_i}, P_{A_j}) \tag{4.11}$$

4.2 LRO Algorithm for the Community Detection

In this section are described the experiments with proposed LRO algorithm for the community detection made for all the data collections mentioned in previous section. The collections have a different size and different density of graph which represent the network.

4.2.1 Edison

Most of capabilities presented in section 3.4 were prepared under the data collection from Edison. We would explain on this dataset all possibilities of the our algorithm.

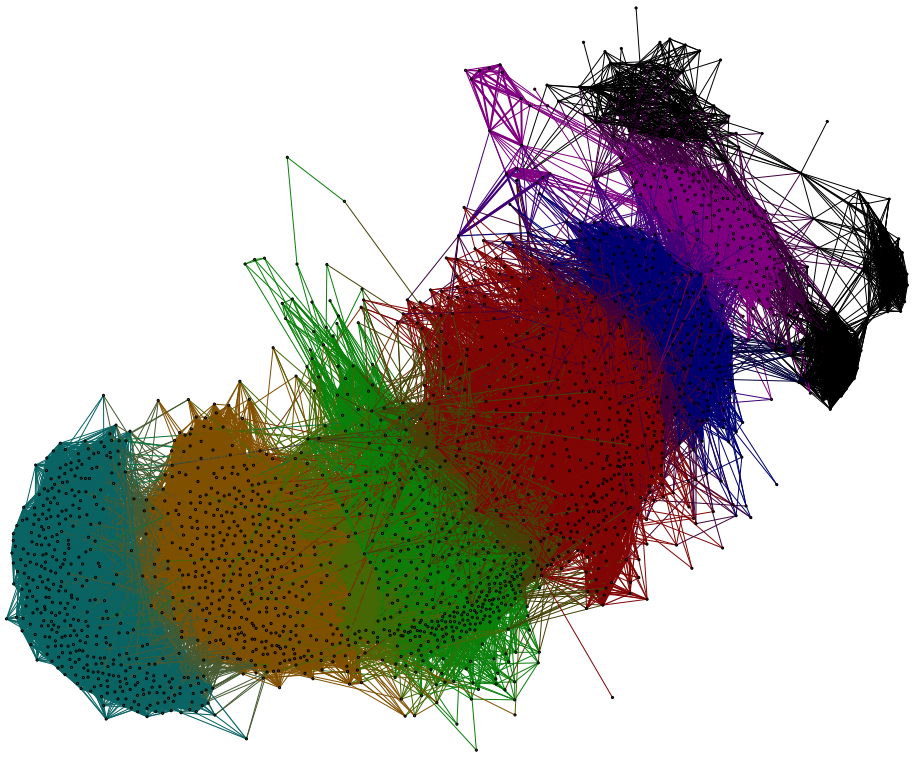

Figure 4.1: Visualization of obtained communities in Edison (LRO algorithm)

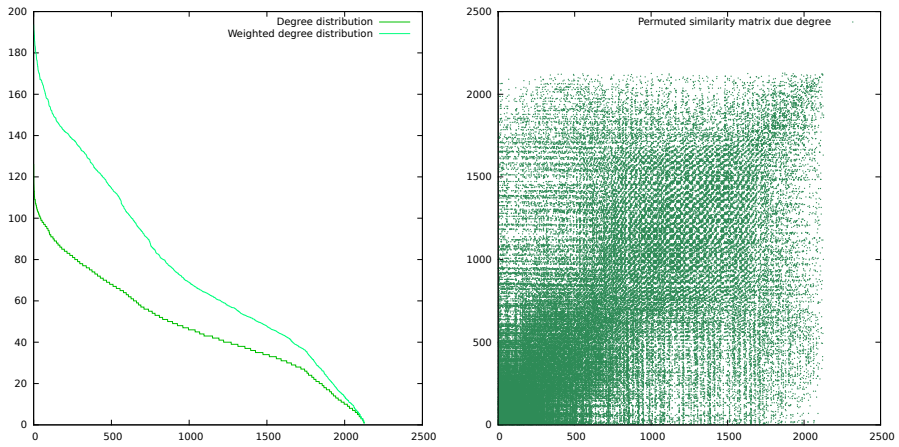

Figure 4.2: Edison: degree distribution and the permuted similarity matrix due the degree

	it=10 S_c=3	it=10 S_c=10	it=100 S_c=3	it=100 S_c=10
Modularity before LRO	0.70199	0.70199	0.71938	0.71938
Modularity after LRO	0.70335	0.70357	0.71973	0.71976
No. of comm. after gaps	41	41	17	18
No. of comm. after LRO	11	8	9	7
Size of comm. 1 after LRO	467	467	436	436
Size of comm. 2 after LRO	409	409	429	429
Size of comm. 3 after LRO	384	390	387	393
Size of comm. 4 after LRO	375	375	378	378
Size of community 5	191	191	186	186

Table 4.4: Different parameters and results for LRO algorithm with Laplacian (Edison)

	S_c=1	S_c=3	S_c=10	S_c=20
Modularity before LRO	0.71938	0.71938	0.71938	0.71938
Modularity after LRO	0.71938	0.71973	0.71976	0.71976
No. of comm. after gaps	17	17	17	17
No. of comm. after LRO	17	9	7	7
Size of comm. 1 after LRO	435	436	436	436
Size of comm. 2 after LRO	429	429	429	429
Size of comm. 3 after LRO	385	387	393	393
Size of comm. 4 after LRO	378	378	378	378
Size of comm. 5 after LRO	185	186	186	186

Table 4.5: Different parameters and results for LRO algorithm with matrix $D^{-1/2}WD^{-1/2}$ (Edison)

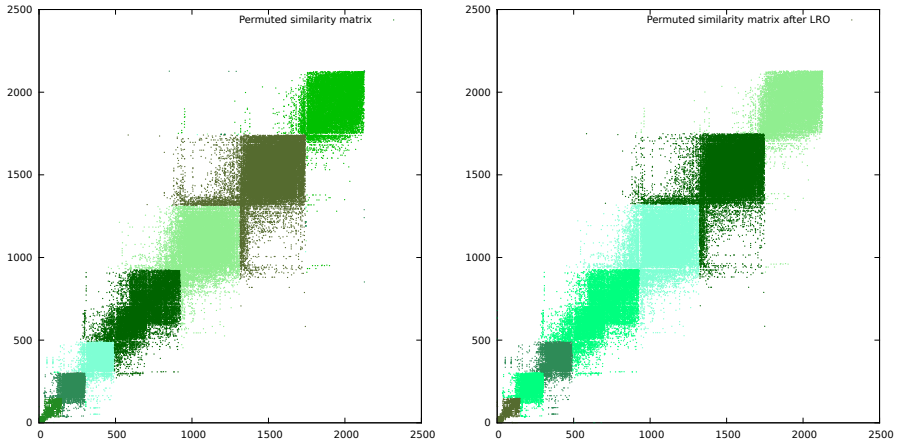

Figure 4.3: Permuted similarity matrix after gaps; permuted similarity matrix after LRO algorithm for Edison

	Similarity>0.8	Similarity>0.7
Number of vertices	5908	5908
Number of edges	32528	69337
Number of isolated vertices	1936	1019
Average Degree	11.01	23.45
Average Weighted Degree	9.674	18.969
Number of connected components	2118	1095
Biggest community - vertices	3017	4643
Biggest community - edges	30181	69093
Average Degree BC	20	29.76
Average Weighted Degree BC	17.577	24.05
Average path length BC	12.36	6.5
Diameter BC	39	20
Size of 2nd and 3rd community	167,62	15,11

Table 4.6: Characteristics of Moodle with the different level of threshold for the similarity

4.2.2 Moodle

For Moodle data collection is characteristic, that we have obtained large amount of different types of relations between students. The evaluation has been computed for all pairs of objects and so the similarity matrix was full. We have used threshold for sparsification of similarity matrix (for example similarity > 0.8).

From the table 4.6 suggests, that most of other communities are very small.The biggest community represents a most part of Moodle. The community detection is useful for this biggest community (see figure 4.4)

Moodle is not typical social network. Its degree distribution (see the table 4.5) does not have a "long tail". In our previous work was important detect a communities of students with a similar patterns of behaviour [78, 93]. Our LRO algorithm is for this situation very suitable.

We can compare the results of our LRO algorithm in the table 4.7 and table 4.8. It is obvious, that the number of edges complicated the situation. The modularity goes down and the biggest community grow. Two parameters in our LRO algorithm are number of iteration in the Lanczos method and the size of the smallest detected community. The table 4.7, the table 4.8 and the table 4.9 show the importance of the high quality of the computed Fiedler vector.

The modularity depends on the high quality of the Fiedler vector (computed by the Lanczos method) and on the size of smallest community (S_c). We can see in the table 4.10 that optimal S_c for our experiment was 4. This number is different in other networks (see the table 4.12). Our algorithm is constructed such that the size of detected communities depends on S_c.

4.2.3 CodePlex

The CodePlex dataset consists of developers and projects. Relation between developers is realised via common projects. The idea is as same in the Edison. But we extend this idea with the context to selected terms, which are in the project's description. We can compare in the Figure 4.6 how important is the vertex "Microsoft" in the network created from CodePlex. This vertex represent typical hub in the network. It has a lot edges to other vertices and the vertex "Microsoft" connected many communities of the developers.

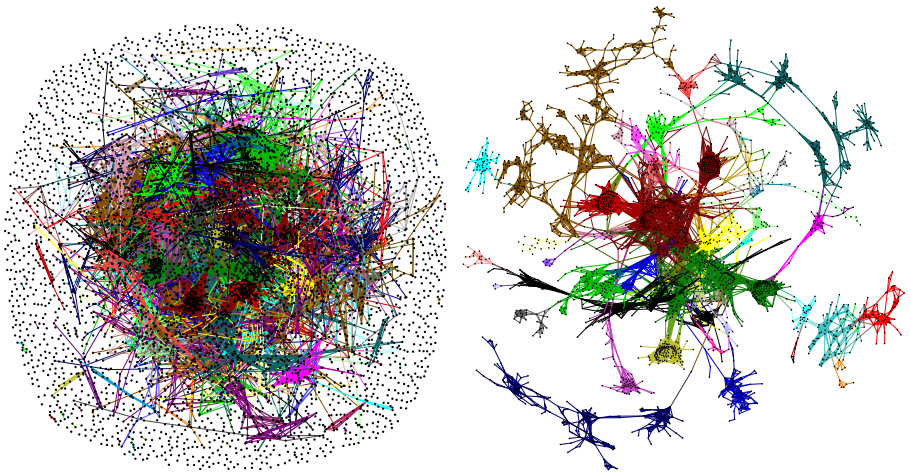

Figure 4.4: Graph of the whole Moodle and the biggest component of Moodle with the similarity > 0.8

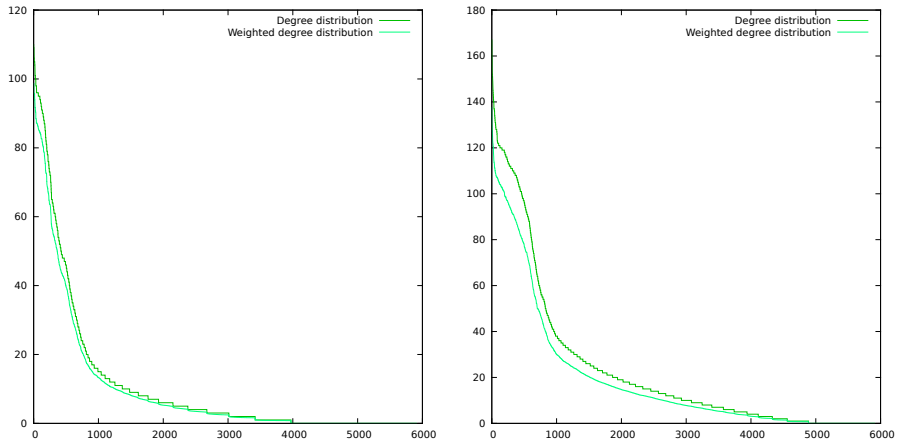

Figure 4.5: Degree distribution of the Moodle with similarity > 0.8 and similarity > 0.7

	it=10 $S_c = 3$	it=200 $S_c = 3$	it=10 $S_c = 6$	it=200 $S_c = 6$
Modularity before LRO	0.6239	0.8159	0.6239	0.8159
Modularity after LRO	0.6508	0.8217	0.6539	0.818
No. of comm. after gaps	2887	2284	2887	2284
No. of comm. after LRO	2373	2270	2445	2398
Size of community 1	972	432	1035	453
Size of community 2	154	417	154	417
Size of community 3	144	327	149	330
Size of community 4	130	214	145	214
Size of community 5	100	190	100	190

Table 4.7: Different parameters and results for LRO algorithm applicated on the dataset Moodle (similarity > 0.8)

	it=10 $S_c = 3$	it=200 $S_c = 3$	it=10 $S_c = 6$	it=200 $S_c = 6$
Modularity before LRO	0.2102	0.4883	0.2102	0.4883
Modularity after LRO	0.2164	0.4962	0.2147	0.4967
No. of comm. after gaps	1737	1490	1737	1490
No. of comm. after LRO	1226	1201	1242	1236
Size of community 1	2803	1431	2886	1451
Size of community 2	648	688	688	687
Size of community 3	197	610	197	620
Size of community 4	118	557	123	556
Size of community 5	115	327	120	349

Table 4.8: Different parameters and results for LRO algorithm applicated on the dataset Moodle (similarity > 0.7)

Number of iteration in Lanczos	10	100	200	300	400	500	600
Modularity before LRO	0.2102	0.5747	0.4883	0.5251	0.5746	0.5821	0.4676
Modularity after LRO	0.2137	0.5904	0.4967	0.5342	0.5910	0.6076	0.4824

Table 4.9: Number of Lanczos iterations and modularity of the detected communities in Moodle (similarity > 0.7)

Size of S_c	1	2	3	4	5	6	10
Modularity before LRO	0.5852	0.5852	0.5852	0.5852	0.5852	0.5852	0.5852
Modularity after LRO	0.5852	0.5979	0.6062	0.6081	0.6075	0.6077	0.6108
Size of community 1	1052	1152	1185	1195	1198	1199	1261
Size of community 2	469	505	527	538	547	547	546
Size of community 3	355	370	372	375	379	379	420
Size of community 4	231	244	250	256	256	256	260
Size of community 5	204	224	226	236	246	254	252

Table 4.10: Modularity and size of the smallest community (S_c) in Moodle for 500 iteration in the Lanczos method

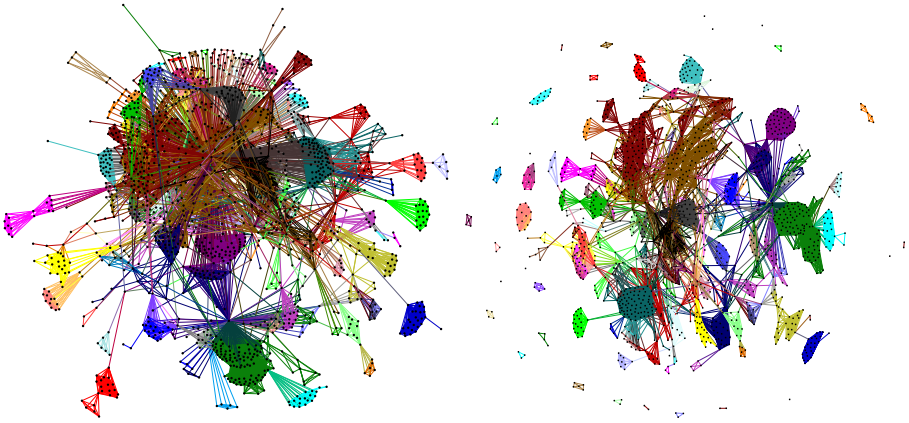

Figure 4.6: CodePlex with vertex "Microsoft"; CodePlex without vertex "Microsoft"

	Whole CodePlex	CodePlex without "Microsoft"
Number of vertices	1463	1462
Number of edges	15318	14587
Average Degree	20.941	19.955
Average Weighted Degree	0.031	0.031
Number of connected components	1	31
Average path length	3.373	5.156
Diameter	10	14
Graph Density	0.014	0.014

Table 4.11: Characteristics of CodePlex

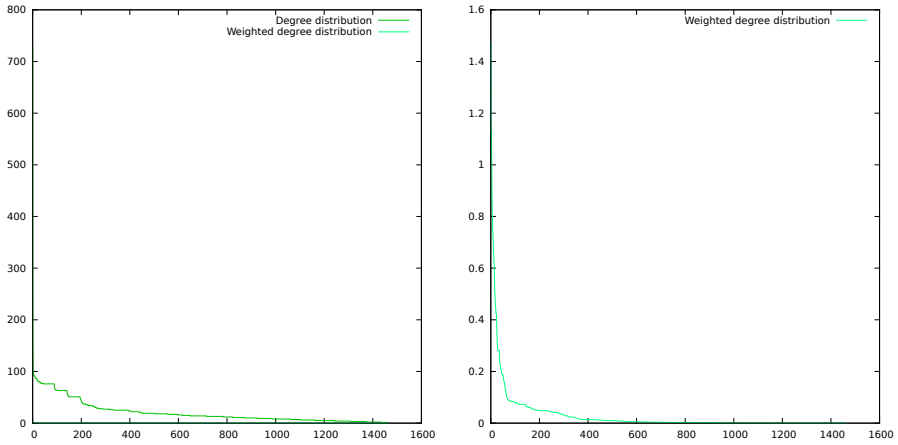

Figure 4.7: Degree and weighted degree distribution of CodePlex

	it=10 $S_c = 3$	it=100 $S_c = 3$	it=10 $S_c = 6$	it=100 $S_c = 6$
Modularity before LRO	0.4577	0.6544	0.4577	0.6544
Modularity after LRO	0.4778	0.6602	0.4914	0.6528
No. of comm. after gaps	392	263	392	264
No. of comm. after LRO	129	99	54	53
Size of community 1	365	175	441	257
Size of community 2	161	129	204	158
Size of community 3	76	122	174	121
Size of community 4	66	95	75	113
Size of community 5	61	59	36	76

Table 4.12: Different parameters and results for LRO algorithm in CodePlex

Degree distribution (see Figure 4.7) is more similar to degree distribution of social network (with power law) than the degree distribution in Edison or Moodle (see the table 4.5 and the table 4.2). But even this network does not "power law".

The table 4.12 describe the dependency of Algorithm LRO on the size of the smallest community and number of the iterations in Lanczos method. These parameters depend on the network structure. We can see in the figure 4.8 different results of algorithm LRO for different values of the size of the smallest component. When is S_c small we obtain more communities (see Table 4.12). We obtain less communities for bigger S_C. In this case is obvious, that $S_c = 20$ is too large because the modularity is smaller than after gaps.

We create a subnetworks of CodePlex network, which are specify by their context to terms - "social", "social network" and "social network database". These terms are considered in the disjunction (i.e. "social" or "network") or in the conjunction (i.e. "social" and "network") (see Figure 4.9).

4.2.4 DBLP

The figure 4.15 and figure 4.13 are a zoomed part of the figure 4.14. The visualization of the whole DBLP is very difficult. But our approach allows a hierarchical point of

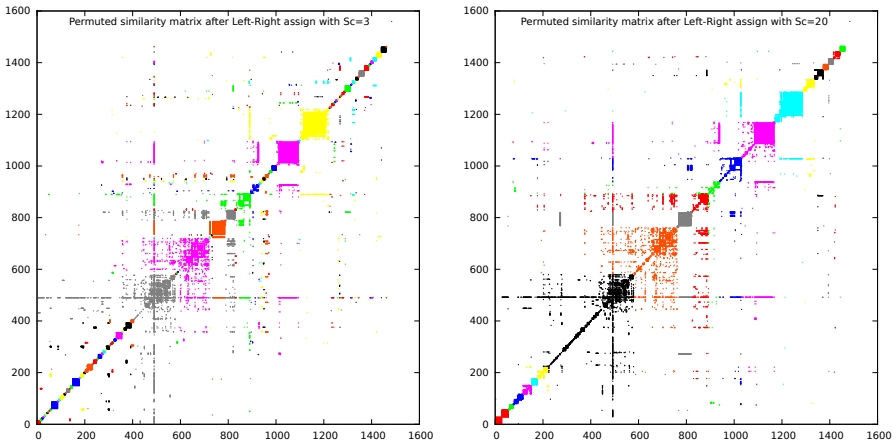

Figure 4.8: How depend algorithm quality on the size of S_c: modularity after gaps=0.6544, modularity after LRO algorithm (with S_c=3)=0.6602 and modularity after LRO algorithm (with S_c=20)=0.6503

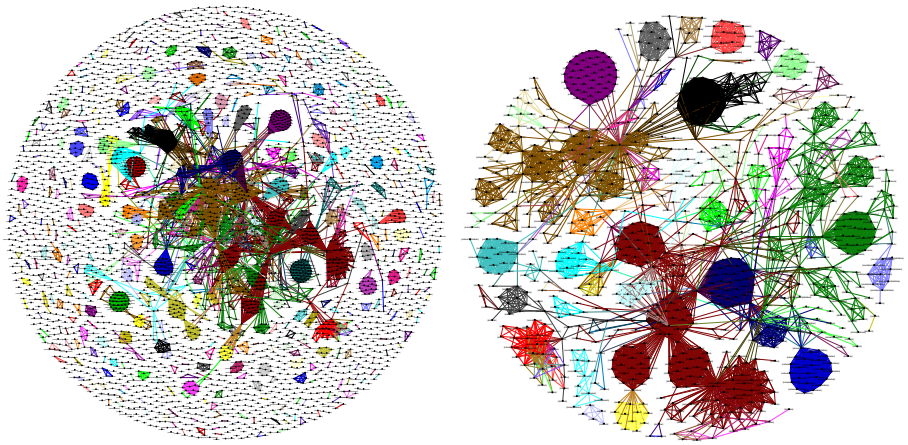

Figure 4.9: CodePlex: derived subnetwork with the context to the ("social" or "network" or "database") and the biggest component in this subnetwork

	$S \vee N \vee D$	BC in SND	$S \vee N$	BC in SN
Number of vertices	3891	1042	1559	333
Number of edges	13489	7768	14522	2865
Average Degree	6.933	14.91	9.315	17.207
Average Weighted Degree	0.021	0.018	0.047	0.041
Number of connected components	1577	1	572	1
Number of isolated vertices	1128	0	402	0

Table 4.13: Characteristics of two subnetworks in CodePlex – context to ("social" or "network" or "database") and context to ("social" or "network")

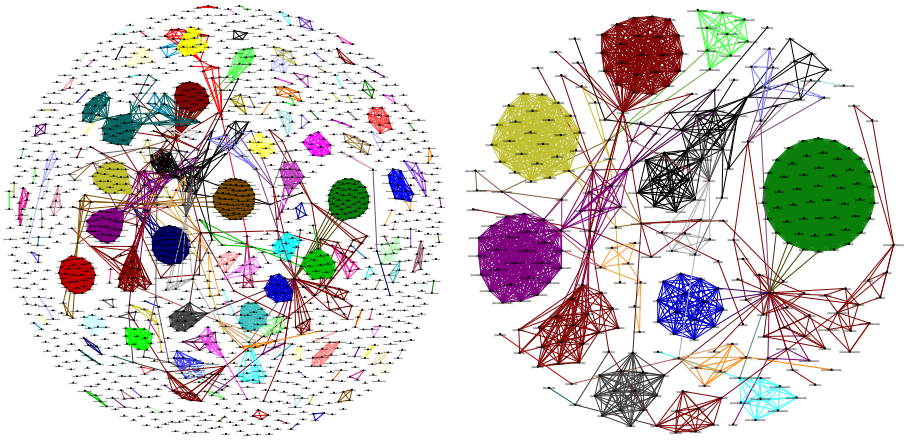

Figure 4.10: CodePlex: derived subnetwork with the context to the ("social" or "network") and the biggest component in this subnetwork

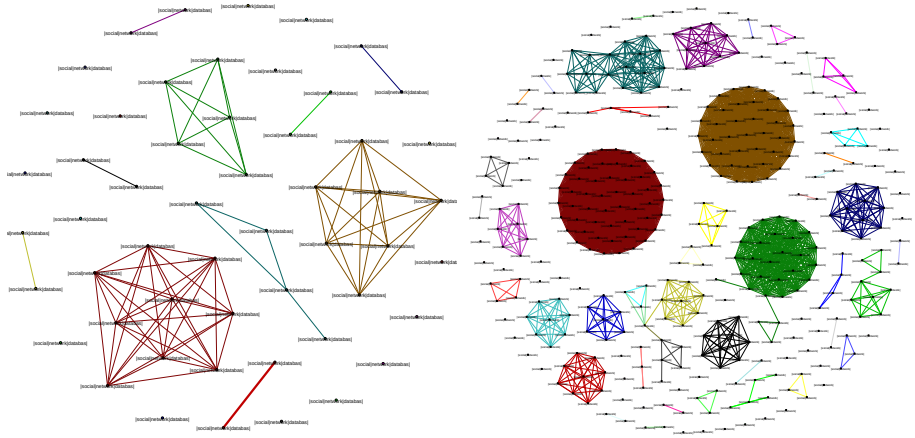

Figure 4.11: CodePlex: derived subnetwork with the context to the ("social" and "network" and "database") and with the context to the ("social" and "network")

	$S \wedge N \wedge D$	BC in SND	$S \wedge N$	BC in SN
Number of vertices	53	9	372	53
Number of edges	154		5800	
Average Degree	6.933	14.91	9.315	17.207
Average Weighted Degree	0.021	0.018	0.047	0.041
Number of connected components	26	1	103	1
Number of isolated vertices	16	0	59	0

Table 4.14: Characteristics of two subnetworks in CodePlex – context to ("social" and "network" and "database") and context to ("social" and "network")

Number of vertices	1109512
Number of edges	7953382
Number of connected components	100067
Number of isolated vertices	55184
Size of biggest community	919375
Number of articles	1914840
Number of extracted terms	1739421
Average Degree	7.16836
Average Weighted Degree	2.95724

Table 4.15: Characteristics of DBLP

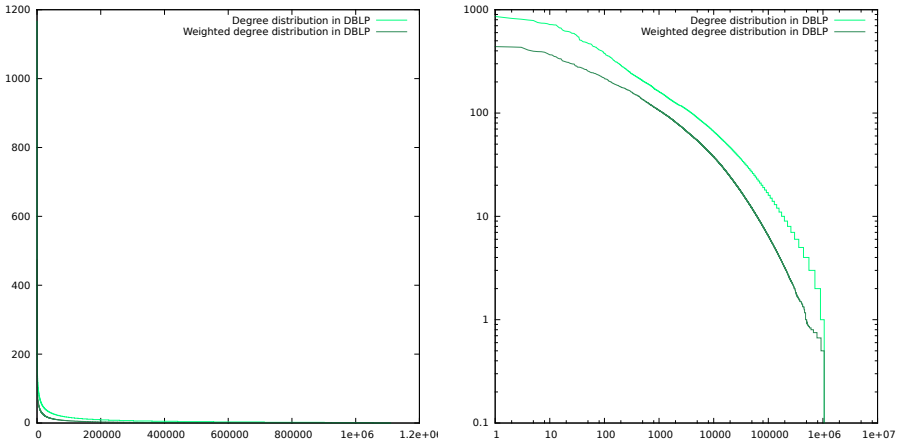

Figure 4.12: Degree distribution of DBLP (normal and log scale)

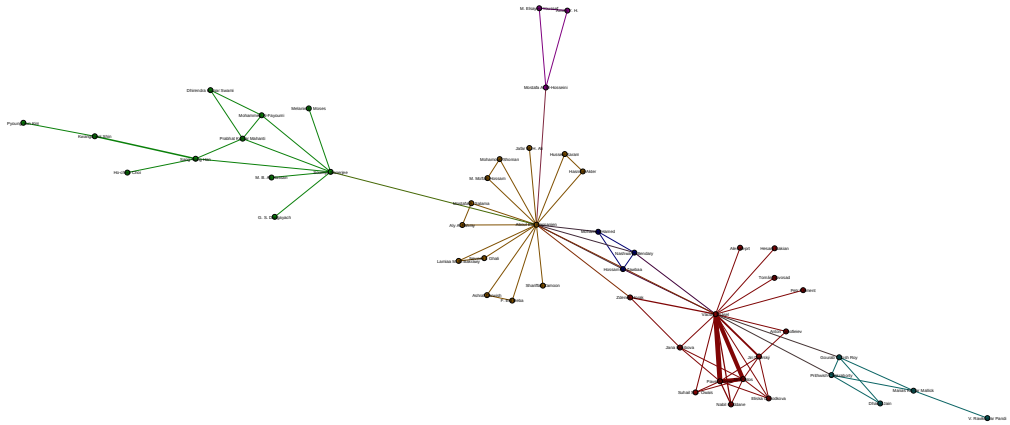

Figure 4.13: Selected community (level 9 in the hierarchical LRO algorithm)

	$S \wedge N$	$Sp \wedge C$
Number of vertices	3444	287
Number of edges	18849	820
Average Degree	5.47	2.857
Average Weighted Degree	0.0069	0.07
Number of isolated vertices	78	10
Number of connected components	318	65
Size of biggest components	2606	102

Table 4.16: Characteristics of two subnetworks in DBLP – context to ("social" and "network") and context to ("spectral" and "clustering")

view to the network. Then, we can select a part of the network with people who are interesting for us.

We can use the same methods of community detection for finding a subnetwork in the original co-authors (collaborators) network, which is determined by selected terms, for which subnetwork edges are evaluated. To be more exact, we mean that the collaborators are in context with the selected terms. Here, the selected terms are not used as a filter, but also as context that evaluates the relation between the co-authors (collaborators). We selected the terms which are interesting for us. Now, we can detect the community publishing in selected area.

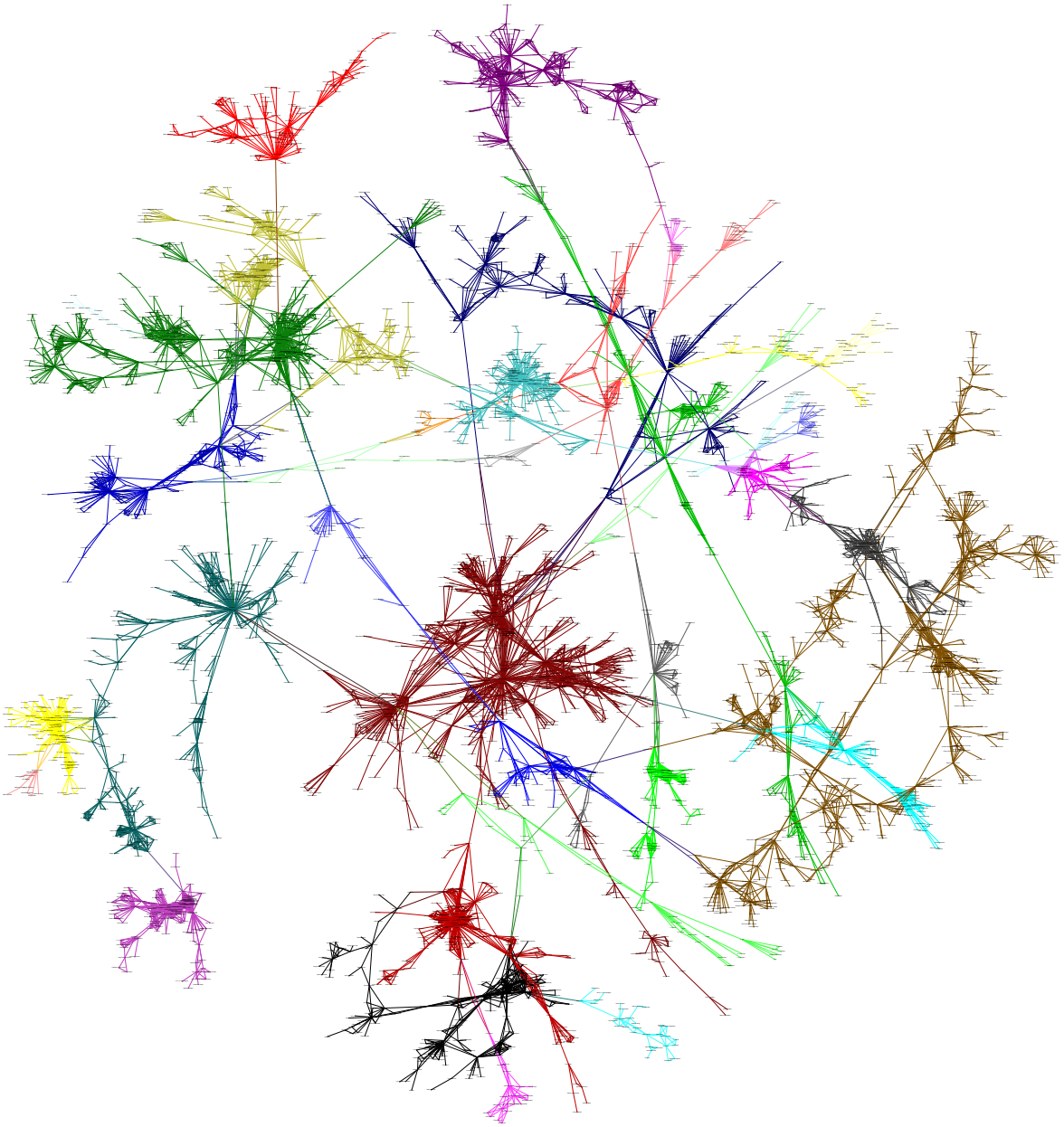

Figure 4.14: Selected community in DBLP with my co-authors (level 7 in the hierarchical LRO algorithm)

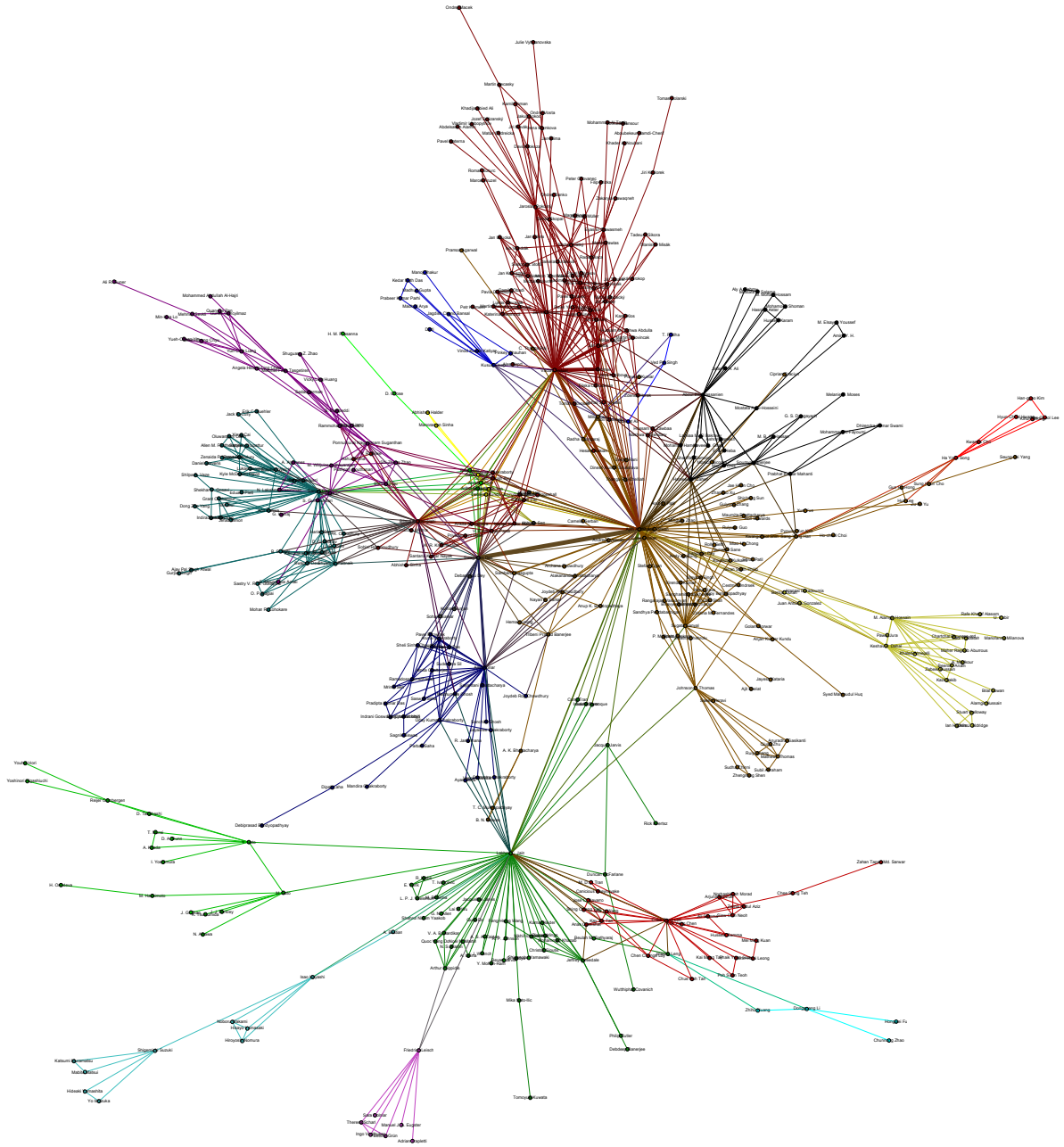

Figure 4.15: Selected community with my co-authors (level 8 in the hierarchical LRO algorithm)

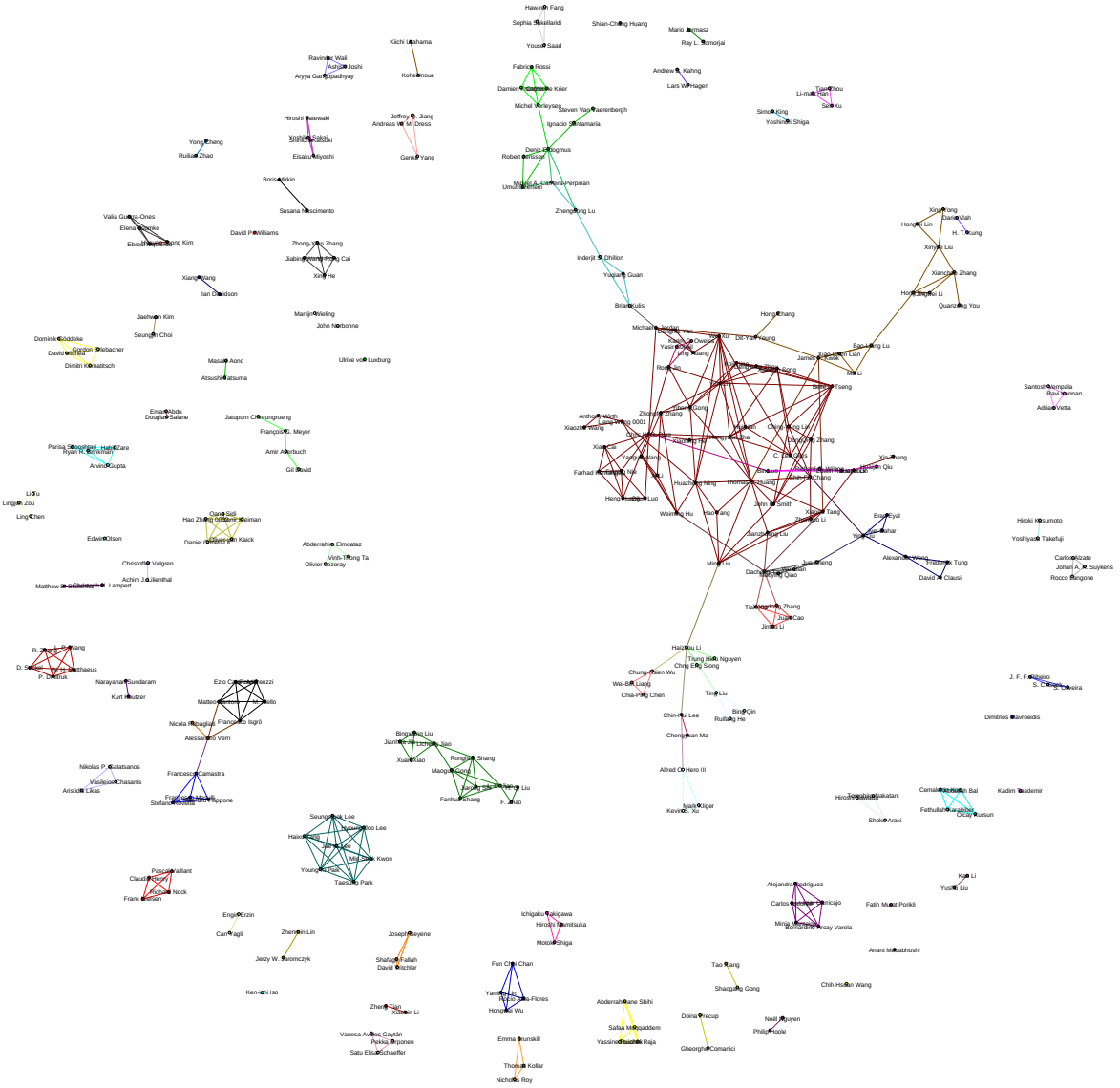

Figure 4.16: Subnetwork in DBLP with context to ("spectral" and "clustering") – communities detected via LRO algorithm

CHAPTER V

Conclusion

In this work, we have addressed the issue of community detection using two different methods. In the first approach, we searched for subnetworks in the original network that maintained the context of selected terms. With this method, we assign all communities that are within our specified context to the original network of co-authors or collaborators on specific projects. This approach is not in line with conventional methods used in searching for communities, but it proved to be an appropriate approach for finding communities that are more precisely specified, described, and in some way connected. Communities found in this way may be directly addressed with a targeted offer (in a network of co-authors it may be an offer for specialized conferences and workshops, or for project collaborators it may be an offer to participate in specific projects within the field of work in which the given developers are experienced). We published about this approach in our papers [65, 64].

The second approach to detecting communities is categorized as spectral clustering algorithms. We used spectral clustering in our papers [97, 78, 95]. In the first phase of our approach, we find the subgraphs using a spectral ordering that is linear. In our work, we used ordering for a given min–cut that we obtain, such as the Fiedler vector Laplacian ($L = D - W$), as well as ordering of a given normalized cut that we obtain, such as an eigenvector corresponding to the second smallest eigenvalue from a symmetric matrix, derived from a similarity matrix ($L_{Sym} = D^{-1/2}WD^{-1/2}$). The second algorithm phase then uses the properties of spectral ordering, which re-assigns nodes to indexes based on their proximity (similar nodes to nearby indexes and less similar nodes to further indexes). In the Left-Right phase (Left-Right-Oscillate) we proceed to spectral ordering with gaps, which are separated into subsets of nodes. In this phase, subgraphs with low connectivity whose nodes were separated into subsets of nodes that did not have connectivity, are re-assigned to nearby subsets with connectivity — future communities. This process allows us to identify the natural amount of communities. For large networks such as the DBLP network we can use a hierarchical method. This approach allows us to increase modularity for found communities in the original, connected network. Surprisingly, this method is more effective with min-cut ordering than with normalized-cut ordering. Our experiments proved that min-cut ordering has worse modularity than normalized-cut ordering, but the Left-Right-Oscillate phase increases modularity for min-cut ordering more. The algo-

rithm we designed captures elements (people), which definitely belong to overlapping communities, more naturally.

Our first approach allows detection of significant, interesting, closely-specified communities of people that we may more easily address and offer cooperation or activities focused on their specialization. Our second approach then allows us to create a community structure within large, complex networks.

The joining of these two approaches may benefit project planners in search of a specific amount of developers in a given field. They may select a rather large community of developers from a subnetwork of experts that they obtained using our Left-Right-Oscillate algorithm, and then contact them with an offer for cooperation.

INVESTICE DO ROZVOJE VZDĚLÁVÁNÍ

Poděkování

Tato práce byla vypracována s podporou projektu Bio-inspirované metody: věda, vzdělávání a transfer znalostí, reg. č. CZ.1.07/2.3.00/20.0073 podpořeného Operačním programem Vzdělávání pro konkurenceschopnost, financovaného ze strukturálních fondů EU a státního rozpočtu ČR.

Bibliography

[1] L. Adamic and E. Adar. Friends and neighbors on the web. *Social Networks*, 25(3):211–230, 2003.

[2] L. A. Adamic, O. Buyukkokten, and E. Adar. A social network caught in the web. *First Monday*, 8(6):1–22, 2003.

[3] L. Ahmedi, L. Abazi-Bexheti, and A. Kadriu. A uniform semantic web framework for co-authorship networks, 2011.

[4] J. Bagrow and E. Bollt. A local method for detecting communities. *Physical Review E - Statistical, Nonlinear and Soft Matter Physics*, 72(4 Pt 2):046108, 2004.

[5] A.-L. Barabasi and R. Albert. Emergence of scaling in random networks. *Science*, 286(5439):11, 1999.

[6] M. Bastian, S. Heymann, and M. Jacomy. Gephi: An open source software for exploring and manipulating networks. 2009.

[7] M. Biryukov and C. Dong. Analysis of computer science communities based on dblp. *Research and Advanced Technology for Digital Libraries*, 6273:9, 2010.

[8] U. Brandes. A faster algorithm for betweenness centrality. *Journal of Mathematical Sociology*, 25(2):163–177, 2001.

[9] L. Branting. Context-sensitive detection of local community structure. *Social Network Analysis and Mining*, 2011.

[10] A. Bruns. How long is a tweet? mapping dynamic conversation networks on twitter using gawk and gephi. *Information Communication Society*, (December 2011):1–29, 2011.

[11] J. Caverlee and S. Webb. A large-scale study of myspace : Observations and implications for online social networks. *Artificial Intelligence*, 8:36–44, 2008.

[12] J. Chang, J. Boyd-Graber, and D. M. Blei. Connections between the lines: augmenting social networks with text. *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2(1):169–178, 2009.

[13] J. Chen, O. R. Zaiane, and R. Goebel. *Detecting Communities in Large Networks by Iterative Local Expansion*, pages 105–112. Ieee, 2009.

[14] J. Chen, O. R. Zaiane, and R. Goebel. Detecting communities in social networks using max-min modularity. *SDM 2009*, page 978–989, 2009.

[15] D. Cheng, R. Kannan, S. Vempala, and G. Wang. On a recursive spectral algorithm for clustering from pairwise similarities. Technical report, MIT, 2003.

[16] D. Cheng, R. Kannan, S. Vempala, and G. Wang. A divide-and-merge methodology for clustering. *ACM Trans. Database Syst.*, 31(4):1499–1525, December 2006.

[17] C.-C. Chou, K.-H. Yang, and H.-M. Lee. Aefs: Authoritative expert finding system based on a language model and social network analysis. In *Proceedings of the 12th Conference on Artificial Intelligence and Applications (TAAI 2007)*, pages 412–419, 2007.

[18] F. R. K. Chung. *Spectral Graph Theory*, volume 92. American Mathematical Society, 1997.

[19] F. Ciravegna, S. Chapman, A. Dingli, and Y. Wilks. Learning to harvest information for the semantic web. *Learning*, 3053(2):312–326, 2004.

[20] A. Clauset. Finding local community structure in networks. *Physical Review E - Statistical, Nonlinear and Soft Matter Physics*, 72(2 Pt 2):7, 2005.

[21] A. Clauset, M. E. J. Newman, and C. Moore. Finding community structure in very large networks. *Physical Review E*, 70(6):1–6, 2004.

[22] D. Crandall, D. Cosley, D. Huttenlocher, J. M. Kleinberg, and S. Suri. Feedback effects between similarity and social influence in online communities. *Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining KDD 08*, 08:160, 2008.

[23] A. Culotta, R. Bekkerman, and A. Mccallum. Extracting social networks and contact information from email and the web. *Processing*, (d), 2004.

[24] A. Dasgupta, J. Hopcroft, R. Kannan, and P. Mitra. Spectral clustering by recursive partitioning. In *ESA'06: Proceedings of the 14th conference on Annual European Symposium*, pages 256–267. Springer-Verlag, 2006.

[25] H. Deng, I. King, and M. R. Lyu. Formal models for expert finding on dblp bibliography data. *2008 Eighth IEEE International Conference on Data Mining*, pages 163–172, 2008.

[26] M.-M. Deza and E. Deza. *Dictionary of Distances*. Elsevier Science, Amsterdam, The Netherlands, Oct. 2006.

[27] C. Ding and X. He. Linearized cluster assignment via spectral ordering. *Twentyfirst international conference on Machine learning ICML 04*, 21:30, 2004.

[28] C. H. Q. Ding, X. He, H. Zha, M. Gu, and H. D. Simon. A min-max cut algorithm for graph partitioning and data clustering. In *ICDM '01: Proceedings of the 2001 IEEE International Conference on Data Mining*, pages 107–114, Washington, DC, USA, 2001. IEEE Computer Society.

[29] Y. Ding. Scientific collaboration and endorsement: Network analysis of coauthorship and citation networks. *Journal of infometrics 5.1*, pages 187–203, 2011.

[30] W. E. Donath and A. J. Hoffman. Lower bounds for the partitioning of graphs. *IBM Journal of Research and Development*, 17(5):420–425, 1973.

[31] S. N. Dorogovtsev and J. F. F. Mendes. *Evolution of Networks: From Biological Nets to the Internet and WWW*, volume 57. Oxford University Press, 2003.

[32] P. Dráždilová, K. Slaninová, J. Martinovič, G. Obadi, and V. Snášel. Creation of students' activities from learning management system and their analysis. In A. Abraham, V. Snášel, and K. Wegrzyn-Wolska, editors, *IEEE Proceedings of International Conference on Computational Aspects of Social Networks CASON 2009*, pages 155–160, 2009.

[33] P. Eades, Q. Feng, X. Lin, and H. Nagamochi. Straight-line drawing algorithms for hierarchical graphs and clustered graphs. *Algorithmica*, 44(1):1–32, 2005.

[34] P. Eades and Q.-W. Feng. Multilevel visualization of clustered graphs. *Lecture Notes in Computer Science*, Volume 119:101–112, 1997.

[35] J.-P. Eckmann, E. Moses, and D. Sergi. Entropy of dialogues creates coherent structures in e-mail traffic. *Proceedings of the National Academy of Sciences of the United States of America*, 101(40):14333–14337, 2004.

[36] M. Fiedler. Algebraic connectivity of graphs. *Czechoslovak Mathematical Journal*, pages 298 – 305, 1973.

[37] M. Fiedler. A property of eigenvectors of nonnegative symmetric matrices and its application to graph theory. *Czechoslovak Mathematical Journal*, pages 619–633, 1975.

[38] D. Fisher and P. Dourish. Social and temporal structures in everyday collaboration. *Proceedings of the 2004 conference on Human factors in computing systems CHI 04*, 6(1):551–558, 2004.

[39] G. W. Flake, S. Lawrence, C. L. Giles, and F. M. Coetzee. Self-organization and identification of web communities. *Computer*, 35(3):66–70, 2002.

[40] S. Fortunato. Community detection in graphs. *Physics Reports*, 486(3-5):75–174, 2010.

[41] L. Freeman. Centrality in social networks conceptual clarification. *Social Networks*, 1(3):215–239, 1979.

[42] L. Garton, C. Haythornthwaite, and B. Wellman. Studying online social networks. *Journal of Computer-Mediated Communication*, 3(1):0–0, 1997.

[43] M. Girvan and M. E. J. Newman. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences of the United States of America*, 99(12):7821–7826, 2002.

[44] W. Glänzel and A. Schubert. *Analysing scientific networks through co-authorship*, pages 257–298. Kluwer academic publishers, 2004.

[45] M. S. Granovetter. The strength of weak ties mark s. granovetter. 78(6):1360–1380, 1973.

[46] L. Hagen and A. Kahng. Fast spectral methods for ratio cut partitioning and clustering. *IEEE International Conference on ComputerAided Design Digest of Technical Papers*, pages 10–13, 1991.

[47] J. Heer and D. Boyd. Vizster: visualizing online social networks. *IEEE Symposium on Information Visualization 2005 INFOVIS 2005*, 5(page5):32–39, 2003.

[48] N. Henry, A. Bezerianos, and J.-D. Fekete. Improving the readability of clustered social networks using node duplication. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1317–1324, 2008.

[49] N. Henry and J.-d. Fekete. Matlink: Enhanced matrix visualization for analyzing social networks. *Lecture Notes in Computer Science*, 4662(4663):288–302, 2007.

[50] N. Henry, J.-D. Fekete, and M. J. McGuffin. Nodetrix: a hybrid visualization of social networks. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1302–1309, 2007.

[51] D. Húsek, J. Pokorný, H. Řezánková, and V. Snášel. chapter Data Clustering: From Documents to the Web, pages 1–33. 2006.

[52] R. Kannan, S. Vempala, and A. Vetta. On clusterings: Good, bad and spectral. *J. ACM*, 51(3):497–515, May 2004.

[53] D. Knoke and R. S. Burt. Prominence. *Applied Network Analysis.*, pages 195–222, 1983.

[54] D. Knoke and S. Yang. *Social network analysis*, volume 154 of *Quantitative applications in the social sciences*. Sage, 2008.

[55] M. Konchady. *Text Mining Application Programming*. Charles River Media, 1 edition, May 2006.

[56] B. Krause, R. Jäschke, A. Hotho, and S. Gerd. Logsonomy - social information retrieval with logdata. In *Proceedings of the nineteenth ACM conference on Hypertext and hypermedia*, HT '08, pages 157–166, New York, NY, USA, 2008. ACM.

[57] S. H. Lee, P.-J. Kim, Y.-Y. Ahn, and H. Jeong. Googling social interactions: Web search engine based social network construction. *PLoS ONE*, 5(7):11, 2010.

[58] K. H. Leetaru. Culturomics 2.0: Forecasting large-scale human behavior using global news media tone in time and space. *First Monday*, 16(9):2, 2011.

[59] J. Leskovec, K. J. Lang, A. Dasgupta, and M. W. Mahoney. Statistical properties of community structure in large social and information networks. In *Proceeding of the 17th international conference on World Wide Web*, WWW '08, pages 695–704, New York, NY, USA, 2008. ACM.

[60] J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

[61] J. Martinovic, P. Drázdilová, K. Slaninová, and V. Snásel. Relation analysis in elearning. In *CISIM*, pages 133–138, 2008.

[62] Y. Matsuo, J. Mori, M. Hamasaki, T. Nishimura, H. Takeda, K. Hasida, and M. Ishizuka. Polyphonet: An advanced social network extraction system from the web. *Web Semantics Science Services and Agents on the World Wide Web*, 5(4):262–278, 2007.

[63] P. Mika. Flink: Semantic web technology for the extraction and analysis of social networks. *J. Web Sem.*, 3(2-3):211–223, 2005.

[64] S. Minks, J. Martinovic, P. Drázdilová, A. Babskova, and K. Slaninová. Developers' cooperation based on terms of project description. In *DATESO*, pages 38–48, 2012.

[65] S. Minks, J. Martinovic, P. Drázdilová, and K. Slaninová. Author cooperation based on terms of article titles from dblp. In *IHCI2011*, 2011.

[66] B. Mohar. Laplace eigenvalues of graphs – a survey. *Discrete Mathematics*, 109:171–183, 1992.

[67] B. Mohar. Some applications of laplace eigenvalues of graphs. *Methods*, 497:225–275, 1997.

[68] A. Montanari. Community detection via spectral methods. *Signal Processing*, (1), 2011.

[69] J. Mori, Y. Matsuo, M. Ishizuka, and B. Faltings. *Keyword Extraction from the Web for Personal Metadata Annotation*, pages 51–60. Citeseer, 2004.

[70] M. Newman. *Networks: An Introduction*. Oxford University Press, Inc., New York, NY, USA, 2010.

[71] M. E. J. Newman, A.-L. , and D. J. Watts. *The structure and dynamics of networks*, volume 107. Princeton University Press, 2006.

[72] M. E. J. Newman. The structure and function of complex networks. *SIAM Review*, 45(2):58, 2003.

[73] M. E. J. Newman. Coauthorship networks and patterns of scientific collaboration. *Proceedings of the National Academy of Sciences of the United States of America*, 101(Suppl 1):5200–5205, 2004.

[74] M. E. J. Newman. Detecting community structure in networks. *The European Physical Journal B Condensed Matter*, 38(2):321–330, 2004.

[75] M. E. J. Newman. Finding community structure in networks using the eigenvectors of matrices. *Physical Review E - Statistical, Nonlinear and Soft Matter Physics*, 74(3 Pt 2):036104, 2006.

[76] M. E. J. Newman. Modularity and community structure in networks. *Proceedings of the National Academy of Sciences of the United States of America*, 103(23):8577–8582, 2006.

[77] M. E. J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical Review E - Statistical, Nonlinear and Soft Matter Physics*, 69(2 Pt 2):16, 2004.

[78] G. Obadi, P. Dráždilová, J. Martinovič, K. Slaninová, and V. Snášel. Using spectral clustering for finding student's patterns of behavior in social networks. In *Proceedings of the Dateso 2010 Annual International Workshop on DAtabases, TExts, Specifications and Objects*, pages 118–130, 2010.

[79] G. Obadi, P. Drázdilová, J. Martinovic, K. Slaninová, and V. Snásel. Finding patterns of students' behavior in synthetic social networks. In *ASONAM*, pages 411–413, 2010.

[80] K. Okamoto, W. Chen, and X. Y. Li. Ranking of closeness centrality for large-scale social networks. *Frontiers in Algorithmics*, 40(v):186–195, 2008.

[81] J.-P. Onnela, J. Saramaki, J. Hyvonen, G. Szabo, M. A. De Menezes, K. Kaski, A.-L. Barabasi, and J. Kertesz. Analysis of a large-scale weighted network of one-to-one human communication. *New Journal of Physics*, 9(6):25, 2007.

[82] T. Opsahl and P. Panzarasa. Clustering in weighted networks. *Social Networks*, 31(2):155–163, 2009.

[83] S. Phithakkitnukoon and R. Dantu. Inferring social groups using call logs. *Lecture Notes In Computer Science Vol 5333*, 2008.

[84] M. F. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.

[85] A. Pothen, H. D. Simon, and K.-P. Liou. Partitioning sparse matrices with eigenvectors of graphs. *SIAM J. Matrix Anal. Appl.*, 11(3):430–452, July 1990.

[86] F. Radicchi, C. Castellano, F. Cecconi, V. Loreto, and D. Parisi. Defining and identifying communities in networks. *Proceedings of the National Academy of Sciences of the United States of America*, 101(9):2658–2663, 2004.

[87] G. Sabidussi. The centrality index of a graph. *Psychometrika*, 31(4):581–603, 1966.

[88] B. Saha and L. Getoor. *Group Proximity Measure for Recommending Groups in Online Social Networks*. 2008.

[89] J. M. Saleh Alwahaishi, V. Snasel, and M. Kudelka. Analysis of the dblp publication classification using concept lattices. 2011.

[90] G. Salton and C. Buckley. Term-weighting approaches in automatic text retrieval. *Information Processing and Management*, 24(5):513–523, 1988.

[91] R. R. T. Santamaría. *Overlapping Clustered Graphs: Co-authorship Networks Visualization*, pages 190–199. 2008.

[92] S. Schaeffer. Graph clustering. *Computer Science Review*, 1(1):27–64, 2007.

[93] K. Slaninová, R. Dolák, M. Miškus, J. Martinovič, and V. Snášel. User segmentation based on finding communities with similar behavior on the web site. In *Proceedings - 2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology - Workshops, WI-IAT Workshops 2010*, pages 75–78, 2010.

[94] K. Slaninová, T. Kocyan, J. Martinovic, P. Drázdilová, and V. Snásel. Dynamic time warping in analysis of student behavioral patterns. In *DATESO*, pages 49–59, 2012.

[95] K. Slaninová, J. Martinovič, T. Novosád, P. Dráždilová, L. Vojáček, and V. Snášel. Web site community analysis based on suffix tree and clustering algorithm. In *Proceedings - 2011 IEEE/WIC/ACM International Joint Conferences on Web Intelligence and Intelligent Agent Technology - Workshops, WI-IAT 2011*, pages 110–113, 2011.

[96] W. M. P. van der Aalst, H. A. Reijers, and M. Song. Discovering social networks from event logs. *Computer Supported Cooperative Work (CSCW)*, 14:549–593, 2005. 10.1007/s10606-005-9005-9.

[97] L. Vojácek, J. Martinovic, K. Slaninová, P. Drázdilová, and J. Dvorský. Combined method for effective clustering based on parallel som and spectral clustering. In *DATESO*, pages 120–131, 2011.

[98] U. von Luxburg and M. Planck. A tutorial on spectral clustering. *Statistics and Computing*, 17:395–416, 2006.

[99] K. Wakita and T. Tsurumi. Finding community structure in mega-scale social networks. *Analysis*, 105(2):9, 2007.

[100] X. Wang, A. Mccallum, and X. Wei. Topical n-grams: Phrase and topic discovery, with an application to information retrieval. *Seventh IEEE International Conference on Data Mining ICDM 2007*, 0:697–702, 2007.

[101] S. Wasserman and K. Faust. *Social Network Analysis: Methods and Applications (Structural Analysis in the Social Sciences)*. Cambridge University Press, 1994.

[102] D. J. Watts and S. H. Strogatz. Collective dynamics of "small-world" networks. *Nature*, 393(6684):440–442, 1998.

[103] B. Wellman. From little boxes to loosely-bounded networks : The privatization and domestication of community university of toronto. *Social Science*, (October 1998):94–114, 1999.

[104] S. White and P. Smyth. A spectral clustering approach to finding communities in graphs. *Proceedings of the fifth SIAM international conference on data mining*, 119:274, 2005.

[105] Z. Wu and R. Leahy. An optimal graph theoretic approach to data clustering: Theory and its application to image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(11):1101–1113, 1993.

[106] P. M. Zadeh, M. Mohi, and M. S. Moshkenani. Mining social network for extracting topic of textual conversations. *Proceedings of the 5th international conference on Soft computing as transdisciplinary science and technology CSTST 08*, page 232, 2008.

[107] M. Zarei and K. A. Samani. Eigenvectors of network complement reveal community structure more accurately. *Physica A: Statistical Mechanics and its Applications*, 388(8):1721–1730, 2009.

[108] J. Zhang, M. S. Ackerman, and L. Adamic. Expertise networks in online communities: Structure and algorithms. In *WWW '07: Proceedings of the 16th International Conference on World Wide Web*, pages 221–230, New York, NY, USA, 2007. ACM.

[109] J. Zhang, J. Tang, and J. Li. Expert finding in a social network. *Advances in Databases Concepts Systems and Applications*, 4443:1066–1069, 2007.

[110] M. E. Zorrilla, E. M. Ruiz, D. Marín, E. Mora, and J. Segovia. Web usage mining project for improving web-based learning sites. In *EUROCAST*, pages 205–210, 2005.

[111] K. A. Zweig and M. Kaufmann. A systematic approach to the one-mode projection of bipartite graphs. *Social Network Analysis and Mining*, 1(3):187–218, 2011.

5.1

Author's bibliography

International journal

1. V. Snášel, A. A. J. Martinovič, P. Dráždilová, K. Slaninová, T. Daradoumis, F. Xhafa, and A. Martiínez-Monés. E-assessment of individual and group learning processes. *Journal of Computational and Theoretical Nanoscience*, volume 9, January, pp 286–303, 2012, ISSN 1546-1955.

International conference proceedings

1. S. Minks, J. Martinovič, P. Dráždilová, A. Babsková, and K. Slaninová. Developers' cooperation based on terms of project description. In *Proceedings of Databases, Texts, Specifications, and Objects – DATESO2012*, pages 38–48, 2012.
2. K. Slaninová, T. Kocyán, J. Martinovič, P. Dráždilová, and V. Snášel. Dynamic time warping in analysis of student behavioral patterns. In *Proceedings of Databases, Texts, Specifications, and Objects – DATESO2012*, pages 49–59, 2012.
3. K. Slaninová, J. Martinovič, T. Novosád, P. Dráždilová, L. Vojáček, and V. Snášel. Web site community analysis based on suffix tree and clustering algorithm. In *Proceedings - 2011 IEEE/WIC/ACM of International Joint Conferences on Web Intelligence and Intelligent Agent Technology - Workshops, WI-IAT 2011*, pages 110–113, 2011.
4. L. Vojáček, J. Martinovič, K. Slaninová, P. Dráždilová, and J. Dvorský. Combined method for effective clustering based on parallel som and spectral clustering. In *Proceedings of Databases, Texts, Specifications, and Objects – DATESO2011*, pages 120–131, 2011.
5. P. Scherer, M. Vicher, P. Dráždilová, J. Martinovič, J. Dvorský, and V. Snásel. Using svm and clustering algorithms in ids systems. In *Proceedings of Databases, Texts, Specifications, and Objects – DATESO2012*, pages 108–119, 2011.
6. S. Minks, J. Martinovič, P. Dráždilová, and K. Slaninová. Author cooperation based on terms of article titles from dblp. In *International Conference IHCI2011*, 2011.
7. G. Obadi, P. Dráždilová, L. Hlaváček, J. Martinovič, and V. Snášel. A tolerance rough set based overlapping clustering for the dblp data. In *Proceedings - 2011 IEEE/WIC/ACM of International Joint Conferences on Web Intelligence and Intelligent Agent Technology - Workshops, WI-IAT 2011*, pages 57–60, 2010.
8. G. Obadi, P. Dráždilová, J. Martinovič, K. Slaninová, and V. Snášel. Finding patterns of students' behavior in synthetic social networks. In *Proceedings of International Conference on Advances in Social Networks Analysis and Mining - ASONAM2010*, pages 411–413, 2010.
9. G. Obadi, P. Dráždilová, J. Martinovič, K. Slaninová, and V. Snášel. Using spectral clustering for finding students' patterns of behavior in social networks. In *Proceedings of Databases, Texts, Specifications, and Objects – DATESO2010*, pages 118–130, 2010.
10. P. Dráždilová, K. Slaninová, J. Martinovič, G. Obadi, and V. Snášel. Creation of students' activities from learning management system and their analysis. In A. Abraham, V. Snášel, and K. Wegrzyn-Wolska, editors, *IEEE Proceedings of*

International Conference on Computational Aspects of Social Networks CASON 2009, pages 155–160, 2009.

11. J. Martinovič, P. Dráždilová, K. Slaninová, and V. Snášel. Relation analysis in elearning. In *Proceedings of Internation Conference on Information Systems and Industrial Management CISIM2008*, pages 133–138, 2008.
12. P. Dráždilová, J. Martinovič, K. Slaninová, and V. Snášel. Analysis of relations in elearning. In *Proceedings - 2011 IEEE/WIC/ACM of International Joint Conferences on Web Intelligence and Intelligent Agent Technology - Workshops, WI-IAT 2011* , pages 373–376, 2008.

Book chapters

1. P. Dráždilová, G. Obadi, K. Slaninová, S. A. Al-Dubaee, J. Martinovič, and V. Snášel. Computational intelligence methods for data analysis and mining of elearning activities. In *Computational Intelligence for Technology Enhanced Learning*, pages 195–224, 2010, ISSN 1860-949X.
2. K. Slaninová, J. Martinovič, P. Dráždilová, G. Obadi, and V. Snášel. Analysis of social networks extracted from log files. In *Handbook of Social Network Technologies*, pages 115–146. 2010.